R-2831-AF

Acquisition and Support of Embedded Computer System Software

Malcolm R. Davis, Steven Glaseman, William L. Stanley

September 1981

Rand

PROJECT AIR FORCE

The research reported here was sponsored by the Directorate of Operational Requirements, Deputy Chief of Staff/Research, Development, and Acquisition, Hq USAF, under Contract F49620-82-C-0018. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

ISBN 0-8330-0434-4
LC Card No. 82-15142

Published by The Rand Corporation

R-2831-AF

Acquisition and Support of Embedded Computer System Software

Malcolm R. Davis, Steven Glaseman, William L. Stanley

September 1981

A Project AIR FORCE report
prepared for the
United States Air Force

Rand
SANTA MONICA, CA. 90406

PREFACE

Under the Project AIR FORCE Resource Management Program study "Software Development and Support" Rand has examined the life cycle characteristics of software "embedded in" (an integral part of) major defense systems, such as aircraft or missile weapon systems. The Air Force is involved in acquiring and supporting at least five general categories of embedded computer system software; this study focuses on operational flight programs and automatic test equipment. This report documents the results of work aimed at understanding the software life cycle and at identifying the resources committed to its various phases. It also examines some commonly held ideas about the development and support of embedded computer system software.

Findings and conclusions of this research, initially reported to the Air Force in briefing form in mid-1980, are based on interviews and data collection efforts that occurred between late-1979 and mid-1980. Readers should recognize that although most of the issues addressed in this report are still pertinent, Air Force initiatives undertaken since that time may have ameliorated some of the problems identified here.

The report is heavy with acronyms and other short forms, which to many technically informed readers are a familiar special language. The authors have attempted to lighten the load on the reader who is unfamiliar with these acronyms by including a comprehensive glossary.

These results should be useful to Air Force and other agencies responsible for, or interested in, the acquisition and support of software embedded in major weapon systems.

SUMMARY

More and more Air Force systems use embedded computer systems (ECS) as an integral part of their operation, and each ECS generally includes more software than earlier systems.[1] The Air Force faces a considerable challenge in managing the development and support of this growing body of software. Although the investment may not be large compared with investment in engines or airframes for fighter aircraft, for example, its function is usually critical to mission success.

The research reported here has sought to develop an understanding of how ECS software is acquired and supported, to assess the characteristics of the ECS software support task today and in the near future, and to illuminate emerging issues and possible problem areas. We examine these subjects primarily in the context of operational flight programs (OFPs) and automatic test equipment (ATE), which occupy about three-quarters of Air Force Logistics Command (AFLC) manpower devoted to ECS software support. However, we have also surveyed some aspects of the aircrew training device (ATD) software acquisition and support process. The report draws mainly on discussions and data collected at visits to Air Force System Program Offices (SPOs) involved in the acquisition of ECS software and the five Air Logistics Centers (ALCs), as well as some discussions with industry officials and a survey of the literature.

---

[1] An embedded computer system consists of the computer equipment and computer programs (software) used as dedicated elements, subsystems, or components of a larger defense system whose major functions extend beyond data processing. ECS software is generally broken down into five categories: (1) operational flight programs, (2) automatic test equipment, (3) aircrew training devices, (4) Command, Control & Communications, and (5) electronic warfare.

To structure the analysis, we examined four frequently asserted perceptions about ECS software acquisition and support that have stimulated Air Force Systems Command (AFSC) and AFLC interest in this subject.

- The Air Force is facing potentially explosive growth in near-term resource requirements for ECS software support.
- More professional programmers are needed to accomplish the support task than the Air Force will be able to attract.
- The widespread use of higher order computer languages (HOLs), instead of assembly languages, might substantially diminish the ECS software support burden.
- ECS software support will consume an increasingly larger proportion of total life-cycle resources.

We have some reservations about the accuracy of these perceptions, or at least about their applicability to all ECS software categories. Our research suggests that:

Individual ALCs with new OFP and ATE software support responsibilities will need a few hundred additional personnel over the next five years or so. Attracting, training, and retaining this number of personnel will be a considerable challenge for the Air Force, but the aggregate growth in personnel across all the ALCs will probably not approach the explosive growth thought inevitable by some.

OFP and ATE software support requires mostly electronic engineering and technician skills. Although the mix of skills supporting ECS software varies across the ALCs depending on the available resources,

necessary professional programming skills can much more readily be acquired on the job than can knowledge about the operation of digital avionics equipment and how it interacts with system software. Each ALC faces the problem of attracting and retaining people possessing these skills.

However desirable, higher order languages (HOLs) will affect only a small fraction of the recurring OFP support effort, which involves a spectrum of activities, only some of which will be beneficially influenced by the use of HOLs instead of an assembly language. HOLs offer considerable advantages in terms of enhanced programmer productivity, ease of training, transparency of software, and potential transferability of software packages, but they should not be regarded as a panacea for the Air Force's growing OFP support burden.

Rough empirical evidence calls into question the contention that software support costs dominate acquisition costs. The manner in which the Air Force contracts for software makes it difficult to estimate acquisition costs with high confidence. Our limited sample disputes the claim that software support costs for ATE are dominating acquisition costs. Comparable information for OFP costs is more speculative, but even for a fairly wide range of assumptions, OFP support costs do not seem to dominate acquisition costs.

Several factors complicate the Air Force's efforts to effectively acquire and support ECS software. They include personnel/skill shortfalls, managerial/organizational anomalies, documentation deficiencies, inadequate data/metrics, the rapid pace of technological advance, and difficulties in testing new generation equipment. These problem areas may form the foundation for future research.

ACKNOWLEDGMENTS

Personnel from the Air Force's five Air Logistics Centers and several System Program Offices of the Air Force System Command's (AFSC) Aeronautical System Division (ASD) generously provided data and insights to support this research effort. Other offices that provided useful comments on various interim oral and written research products of the study included ASD/Engineering, the Air Force Logistics Command/Directorate of Engineering and Computer Resources and the Acquisition Logistics Division, the Deputy for Avionics Control and the Air Force Systems Command/Directorate of Computer Resource Development Policy and Planning.

Despite this considerable assistance, the authors remain responsible for the conclusions of the study and for any errors or misinterpretations that may appear.

CONTENTS

FIGURES

TABLES

GLOSSARY

AFIT	Air Force Institute of Technology
AFLC	Air Force Logistics Command
AFSC	Air Force Specialty Code
AFSC	Air Force Systems Command
AFTEC	Air Force Test and Evaluation Center
AISF	Avionics Integration Support Facility
ALC	Air Logistics Center
ALD	Acquisition Logistics Division of AFLC
ASD	Aeronautical Systems Division of AFSC
ATD	Aircrew Training Devices
ATE	Automatic Test Equipment
BITE	Built in Test Equipment
C3	Command Control and Communications
CPC	Computer Program Component
CPCI	Computer Program Configuration Item
CPIN	Computer Program Identification Number
DACS	Data and Analysis Center for Software
DEPS	Development Engineering Prototype Sites
DID	Data Item Description
DoD	Department of Defense
ECS	Embedded Computer System
ESD	Electronic Systems Division of AFSC
EW	Electronic Warfare
HOL	Higher Order Language

HUD	Head Up Display
INS	Inertial Navigation System
IOT&E	Initial Operational Test and Evaluation
ITA	Interface Test Adapter
LRU	Line Replaceable Unit
MATE	Modular Automatic Test Equipment
OAS	Offensive Avionics System
OFP	Operational Flight Programs
OOALC	Ogden Air Logistics Center
PMRT	Program Management Responsibility Transfer
RADC	Rome Air Development Center
SAALC	San Antonio Air Logistics Center
SM	System Manager
SMALC	Sacramento Air Logistics Center
SPO	System Program Office
SSC	Software Support Center
TCTO	Time Compliance Technical Orders
TRD	Test Requirement Document
UUT	Unit Under Test
WRALC	Warner Robins Air Logistics Center

I. INTRODUCTION

Over their operational lifetime, software products are frequently changed, extended, and modified for one reason or another. The term "software maintenance" is often applied to these actions, but it is a misnomer because maintenance brings to mind the idea of repair or of restoring to original condition by physical refurbishment or replacement. This concept is most properly associated with hardware devices. However, software does not deteriorate in a physical sense; once operational, it performs its functions without wear.

The changes and extensions applied to software are associated with enhancing capability. The term "software life-cycle support" has come into use within the Air Force, and it more adequately describes the situation. Such support can include:

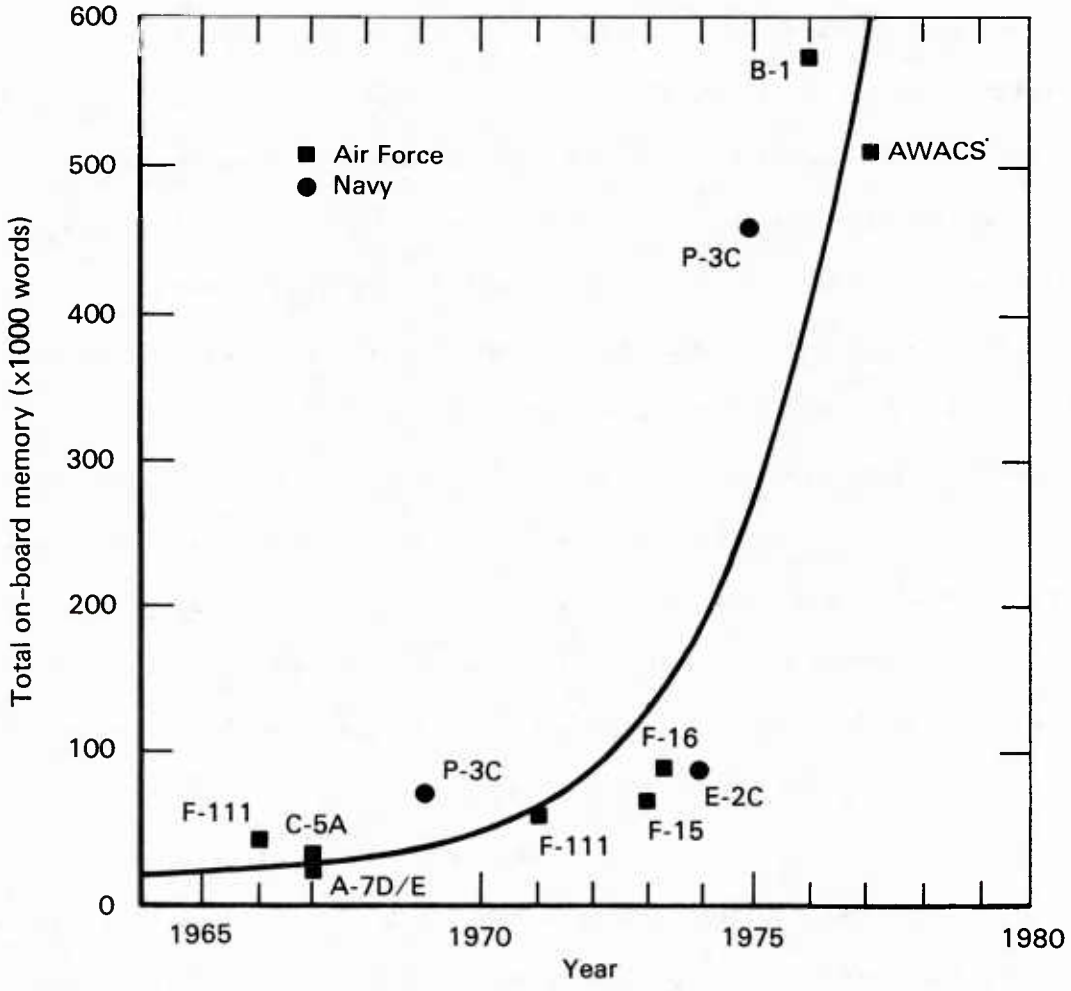
- Correcting program errors remaining from development or introduced in previous change cycles.
- Correcting deficiencies revealed through use.
- Responding to changes in the threat, tactics, or operational environment.
- Adding/modifying software to interface with other subsystems (e.g., a new weapon).
- Deleting capabilities no longer needed.
- Optimizing for more efficient operation.

The level of resources needed to perform this support is of great concern to the Air Force. Historically, most organizations responsible

for acquiring and supporting new weapon systems have not planned for life-cycle support resources as part of the acquisition process. In part, the support requirement was not foreseen in its full scale; in part, there has been insufficient interaction among developers, maintainers, and operators; and in part, the full extent of the life-cycle support resource requirements is extremely difficult to forecast.

To compound the problem, a growing number of Air Force systems are incorporating embedded computer systems (ECS), and each new system generally has more software than the previous system. A commonly used depiction of the trend in growth of total on-board software memory is shown in Fig. 1, which suggests dramatic exponential growth. There are several reasons for this growth:

- Software provides a means to perform functions not previously possible with analog or mechanical devices.
- At lower cost and with greater reliability and flexibility, software also provides the ability to perform functions previously done by analog or mechanical devices.
- When the choice is between changing hardware or software to meet new operational needs, changing the software is usually more expedient. For example, once the change has been designed and a copy of the modified computer program tested, the replication of that program is a fairly efficient, inexpensive, highly automated, and controlled process. Furthermore, changing software usually implies no very great purchase of new parts or components, and the modified programs can be shipped to the field. In contrast, changing hardware means spending the same amount of effort modifying every item of equipment and



Source: AFSC/XR

Fig.1—A common depiction of growth in DoD avionics software

often bringing each item of equipment to the skilled personnel and facilities to make the required changes.

In spite of the increasing reliance on software for weapon system capabilities, the dimensions of software development and support are not yet well understood, well documented, or even a new subject for study and analysis. During the past ten years or so the Air Force has

conducted and supported many studies and workshops to gain a foothold on the problem.[1] Each study contributed a bit of knowledge and understanding, but total understanding is still lacking. This seemingly slow progress is partly because the rate at which complexity has increased in newer systems has kept the number of unsolved problems high, despite the efforts by the Air Force to deal with them.

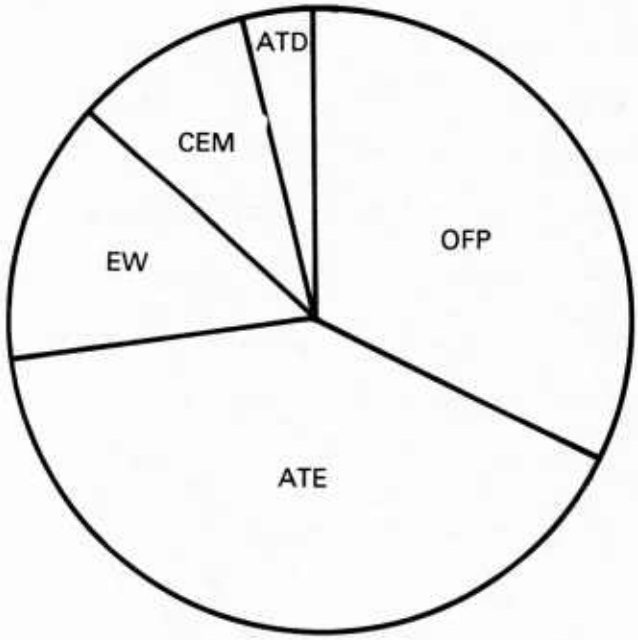
Air Force Systems Command (AFSC), Air Force Logistics Command (AFLC), and the operating commands must deal with at least five different categories of ECS software.[2] This study primarily examines two of these categories, operational flight programs and automatic test equipment. AFLC estimates that 75 percent of its personnel involved in ECS software support deal with these two categories of software (see Fig. 2). There is heavier user involvement in software development and support in command and control and electronic warfare, which are not covered in this study. The study's focus was subsequently broadened somewhat to include attention to some aspects of aircrew training device software. However, we were unable to have discussions with the user community or visit Development Engineering Prototype Sites (DEPS).

Our research was aimed at understanding the nature of the software development and support task, including organizational interactions, the present and emerging support workload, the resource commitments today and in the near future for support, and problem areas and emerging issues with respect to acquisition and support. We interviewed personnel from Air Logistics Centers, System Program Offices, other Air

---

[1] See Refs. 1-10.

[2] Command, control & communication (C3), electronic warfare (EW), aircrew training devices (ATDs), automatic test equipment (ATE), and operational flight programs (OFPs).



Total 1413 people

Source: AFSC/XR

Circa 1978

Fig. 2—Summary of AFLC software support manpower

Force organizations, and certain commercial firms; we also surveyed the relevant literature. By establishing a baseline of conventional wisdom, we identified several broadly (but not universally) held perceptions that have focused Air Force attention on the task of ECS software development and support. This conventional wisdom suggests:

- The Air Force is facing explosive growth in near-term resource requirements for ECS software acquisition and support.
- Software support will consume an increasingly larger portion of total life cycle resources.
- The Air Force will need (and may not be able to attract) many professional programmers to support ECS software.

o The widespread use of higher order languages (HOLs) will substantially reduce the emerging software support burden.

Figure 3 first appeared in 1972 as a rough forecast of the distribution of hardware and software costs for large-scale command and control computer installations.[3] It suggested how the total Air Force budget for command and control computers might be divided between hardware and software. It has become a popular illustration of more general industry trends toward less costly hardware and more costly software, both initial cost and life-cycle support. Since its

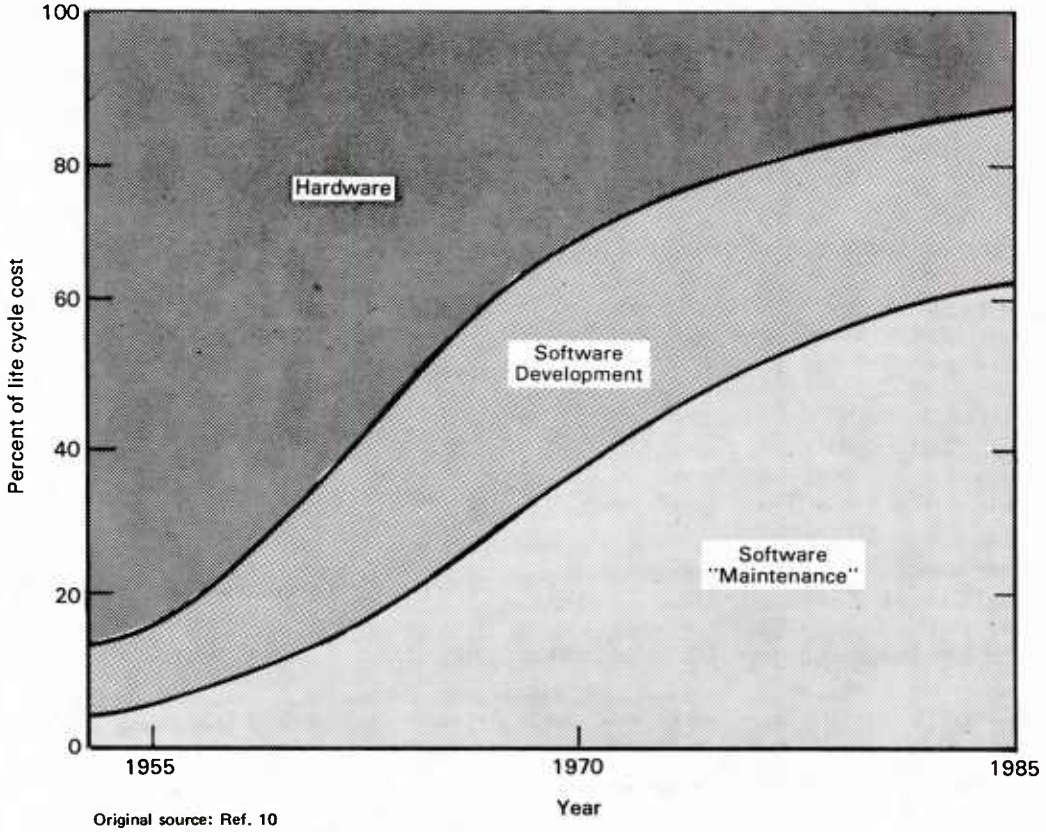


Fig. 3—Common depiction of trends in embedded computer systems hardware and software costs

[3] Ref. 10.

introduction, a new line has been added to denote the fractional contributions of development and maintenance to total software costs.

We have not located any studies that provide data to support such an addition. More worrisome, however, is that frequent exposure and republication seems to have given this figure a credibility beyond that warranted by either its original emphasis or the data upon which it was based. A widely held perception in the Air Force, based largely on this very broad and imprecise figure, is that growing software life-cycle support costs will soon dominate ECS life-cycle cost.

This study assesses the validity of such perceptions using empirical information whenever possible. It is easy to be led astray by failing to recognize that the acquisition and support processes of the five categories of embedded computer systems are very different from each other. This report purposely maintains those distinctions.

II. THE OPERATIONAL FLIGHT PROGRAM DEVELOPMENT AND SUPPORT PROCESS

This section describes the development and support process for operational flight programs, identifies the major resource commitments, and examines the commonly held perceptions relative to ECS development and support. The specific topics addressed include (1) the organizational framework for OFP development and support, (2) support responsibilities and the support process, (3) resource requirements for support, (4) personnel skill requirements, (5) potential influences of higher order languages, and (6) a comparison of acquisition and support costs.

ORGANIZATIONAL FRAMEWORK

Typically, contractors (either prime or sub) develop OFPs under contract to and direction from an AFSC System Program Office (SPO) or AFLC System Manager. The quality of the delivered OFP depends on a number of factors:

- Early consideration of software and software support problems in the system design phase.
- A good and proper set of functional requirements.
- A reasonable and well-considered contract.
- Adequate resources.
- The experience and ability of the contractor.
- The experience and ability of the Program Office personnel.
- The existence and reasonable application of guidance in the form of regulations, directives, and standards.

A previous Rand report has discussed many of these issues, and several Air Force workshops and studies have probed into the problems of developing a high quality OFP.[1]

Figure 4 shows some of the more important organizational interactions that occur during the development and support of OFPs. In the development of new systems, the System Program Office within AFSC normally has the responsibility for managing OFP development, as depicted in Fig. 4. For example, the F-16 SPO manages the development of F-16 OFP software. When major modifications involving new program development are made to existing systems--an increasingly common occurrence as fewer and fewer completely new weapon systems are being acquired--the System Manager under AFLC control may manage the development; for example, the F-111 System Manager at Sacramento Air Logistics Center is managing the computer replacement program for the F-111.

Whoever manages software development, and indeed subsequent support, is subject to various forms of guidance from the organizations shown across the top of Fig. 4. In some instances this guidance is very precise, requiring rigorous compliance; in other cases the guidance is general and subject to specific interpretation by the Program Office.

The SPO decides on the application of the guidance and uses it to review and direct the activities of the contractor to assure a quality and supportable OFP.[2] In some instances the Program Manager is

---

[1] Refs. 6, 9, 11, 12, 13.

[2] Contractors and subcontractors can be involved in many aspects of software development and support. To avoid making Fig. 4 undecipherable, we have illustrated their involvement stylistically.

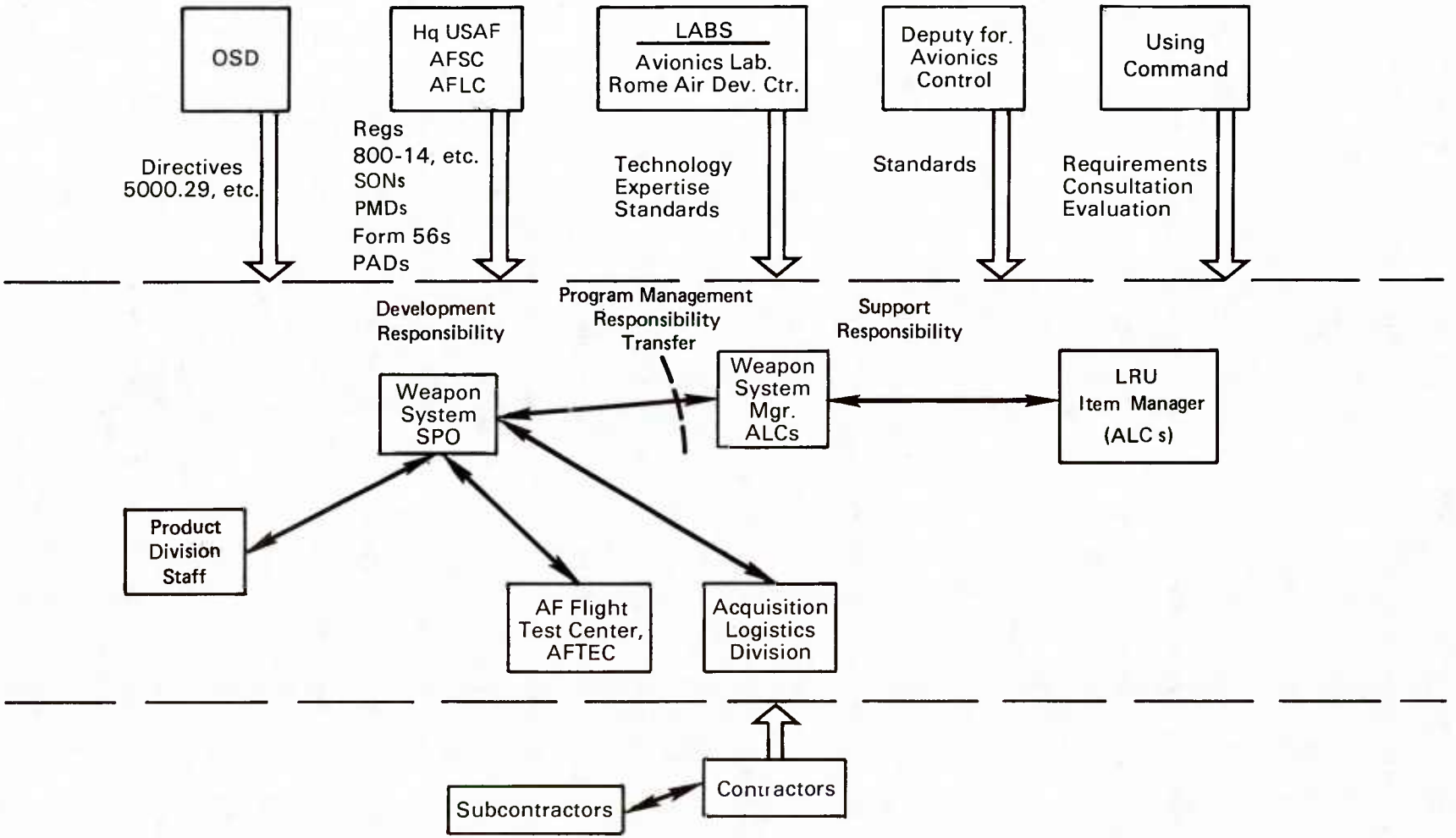


Fig. 4—Major organizations involved in acquisition and support of OFP software

fortunate enough to have the needed expertise and experience on board. In other cases, SPOs may have limited software capability. The limitation is usually a shortage of people with the necessary software expertise and experience. This is sometimes traceable to a funding problem that may be caused by a program funding shortage, by insufficient emphasis placed on software by the Program Manager, or by unrealistic schedules.

The importance of early consideration of software issues may not be recognized, and they may therefore be dealt with only superficially until late in the program. This can result in the ECS being late, unreliable, unsupportable, and less capable than originally intended.

If SPO personnel recognize the importance of software, they frequently seek external assistance, which may be formal or informal and may come from various sources; however, its availability is not always assured. Figure 4, for example, shows that the SPO may ask for assistance from the Acquisition Logistics Division (ALD) of AFLC. However, at the time of our interviews, in an organization of over 1000 people, ALD had only nine people with software experience, and they were being spread over other areas besides OFP development. The SPO may also ask for assistance from various members of the product division staff--e.g., the Aeronautical Systems Division Engineering Directorate (ASD/EN), but here too the numbers of people with experience in initiating and monitoring software development contracts are limited and the demand is great.

In some cases a SPO will seek the expert assistance of another contractor in reviewing and directing a prime contractor's OFP

development efforts. This has often worked well (B-52 offensive avionics system, for example); however, the assistance contractor must have a high level of expertise with a proven track record for this to be a worthwhile approach.

The Air Logistics Centers are an alternative source of assistance to the SPO. Although ALCs do not normally have development responsibilities for new systems, their acceptance of an assistance role before Program Management Responsibility Transfer (PMRT) has proved valuable both to SPOs in helping to produce a quality OFP and to the ALCs themselves in preparing to accept total system support responsibility.[3] However, ALC assistance is not without problems. For example, an ALC that has not yet been assigned system responsibility nor been funded for that task might have neither the people nor other resources available to assist the SPO early in the program when help is often needed most.

The ALC that has been assigned OFP support responsibility for a given program is understandably concerned about the quality of the product it is inheriting, especially relative to supportability. Consequently, responsible ALC personnel may try to influence some aspects of the OFP software in the face of inadequate resources and without being asked. The success of these efforts is varied and dependent on their acceptance by the SPO. As the acquisition process is now organized, the final authority for inclusion or exclusion of any recommendation rests with the SPO.

---

[3] In the case of the F-16, the Ogden ALC and the Air Force Avionics Laboratory did the validation and verification of the OFP software. Feedback to the SPO and the contractor helped improve the product, and the exercise prepared Ogden ALC for their future support role.

The testing agencies depicted in Fig. 4 identify software inadequacies for the SPO and the user during the Development and Operational test and evaluation phases. AFTEC in particular is beginning to play a more important role in the independent validation and verification of OFP software during Operational Test and Evaluation. Policy is still evolving in this area, but in the past, SPO directors have usually had the responsibility and authority to decide whether their system underwent this phase.

After a system has attained a level of operating capability, the responsibility for support is transferred from the SPO to the System Manager at an Air Logistics Center as depicted in Fig. 4. Program management responsibility transfer for the OFP may or may not be concurrent with PMRT for other elements of the system. Moreover, transfer of engineering responsibility for support of an OFP may occur some years before to the transfer of management responsibility.[4]

The System Manager may elect to do the support using Air Force facilities and personnel (organically) or award a support contract. If a contractor is used for support he may or may not perform the support function at an Air Force facility. Often, if there is a shortage of personnel, a lack of training, or extremely complex and specialized subsystems, there is no choice but to use a contractor. To do otherwise would be too risky. This decision, of course, is made some time before the PMRT date.

---

[4] For example, engineering support responsibility for some F-16 software is scheduled to shift to Ogden ALC by October 1981, whereas the system PMRT is planned for October 1985.

Figure 4 illustrates one additional problem facing System Managers that is a consequence of the proliferation of processors on new weapon systems. Although one weapon System Manager is generally responsible for the support of an integrated system, there are instances in which Line Replaceable Units (LRUs) making up that system, each having its own processor and software, are the responsibility of item managers at other ALCs.[5] For example, although Sacramento ALC is responsible for the A-10 system as a whole, ALCs at San Antonio, Ogden, and Warner Robins also have control of individual LRUs. Because a software change in one LRU might affect other parts of the system, including not only OFP but also ATE and ATD software, this poses a formidable communication problem for the System Manager.

SUPPORT RESPONSIBILITIES AND PROCESS

Support responsibility assignments are made well in advance of the PMRT date to allow the Air Logistics Center to plan the support activities, analyze, prepare, and get approval for needed manpower and budget; recruit and train the needed personnel; and design and acquire the support facilities. Each ALC has approached these activities in a slightly different way, depending on past experiences and the individual ALC organization and politics. Some of these differences will be discussed later.

---

[5] The situation is somewhat different for F-16 support at Ogden ALC. Ogden will have responsibility for all LRUs on the multiplex bus, as well as the avionics intermediate shop test equipment software. Recognition of the integrated nature of the weapon system from the start, and the desire of the multinational program participants for a single focal point, led to this arrangement.

Table 1 shows ALC Operational Flight Program support assignments for major systems. Some of these systems are operational and are currently being supported by the assigned ALC, and others are still in the support planning stages with responsibility transfer scheduled for a later date or not yet established. Although the distribution of system responsibilities across the different ALCs[6] may appear unbalanced, Table 1 depicts only OFP support assignments. Also, each system does not require the same level of effort; some are quite modest and some are extensive. However, there are many new system support responsibilities facing the ALCs in the early and mid-1980s.

Table 1
OPERATIONAL FLIGHT SOFTWARE SUPPORT RESPONSIBILITIES

	San Antonio	Sacramento	Ogden	Warner Robins	Oklahoma City
Post PMRT Systems	F-106 C-5A Special wpns	FB-111A (1973)[a] F-111F (1974) F-111D (1975)	Titan II (1964) Minuteman II (1978) F-4E/RF-4C (1978) F-4G (1979)		A-7D/K (1975) SRAM (1976) Navigation systems
Pre PMRT Systems (as of mid-1980)		A-10 (1982)	Minuteman III (1981) F-16 (1985) MX Precision guided munitions	PAVE TACK (1981) F-15 (1982) HARM (1983) EAR (1983) HAST (1984) CMMR (1984) GPS (1986) AMRAAM (1986) JTIDS (1983)	B-52 OAS (1982) ALCM (1982) B-52 DBNS/DCU GLCM E-4A (1982) E-4B

[a] ( ): Program management responsibility transfer date

[6] Under the Technical Repair Center concept, support assignments are generally made based on specialized capability of each ALC, and not on equalization of workloads.

Operational flight programs have some characteristics that must be considered in preparation for support. To begin with, the vehicles they are embedded in have space and weight restrictions that limit the usable computer hardware. We are entering an era in which large, high-density memories and high throughput processors will be available at reduced cost and volume. Although this augurs well for the future, systems whose development began during the 1970s and that are entering the inventory today were designed subject to memory size and throughput constraints. Memory sizes of 64K words or less and speeds of less than 500K operations per second are typical today. Therefore, most OFPs are tightly coded in assembly language in comparison with higher order languages, because the HOL compiler-produced object code is not optimized for throughput and is less efficient in memory space utilization (10 to 15 percent greater memory requirements). Table 2 shows some characteristics of the operational flight software for the F-111 and F-16 systems. Even for a recent system such as the F-16 the majority of operational flight software is still coded in assembly language, partly because some subsystems are procured off the shelf and come with existing assembly language software packages. To have these OFPs reprogrammed in a higher order language and to provide the required additional memory imposes additional costs and development time that SPOs have been unable or unwilling to accept. For these reasons Air Force systems will continue to have operational flight software written in assembly language for several years to come.

Many characteristics of the flight software itself contribute to the complexity of the total software support task.

Table 2

EXAMPLE CHARACTERISTICS OF OPERATIONAL FLIGHT SOFTWARE

System	Program	Memory size K of equivalent 16 bit words	Language
F-111D,F	Navigation/weapon delivery[a]	32	Assembly
	Inertial navigation[a]	3	Assembly
	Total	35	
F-16	Fire control[a]	35	85%JOVIAL(J3B-2), 15% Assembly
	Stores Mgt. System[a]	8	Assembly
	Head up display[a]	8	Assembly
	Fire control radar	36	Assembly
	Inertial Nav. set	12	Assembly
	Radar E-O displays	4	Assembly
	Central Air Data	3	Assembly
	Threat warning[a]	22	Assembly
	Total	128	

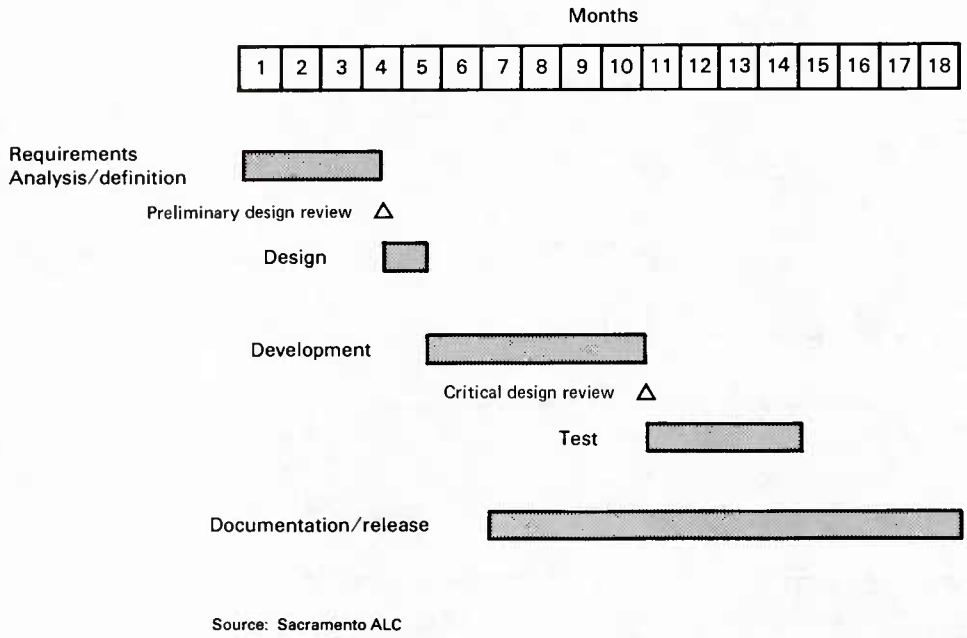
[a](to be) supported "organically".

- o Certain flight software functions can be critical to flight safety. The hangup of a missile or a false flydown signal when flying in a terrain following mode can be disastrous.
- o OFPs must operate in real time.
- o Flight software is closely integrated with the hardware, taking many of its inputs from analog to digital converters and feeding outputs to digital to analog converters.
- o Avionics functions are tightly linked, requiring extensive interchange of information among a number of subsystems.
- o Flight software is often used to compensate for unexpected hardware deficiencies, which can make it more complicated.

The Air Force organization having the most experience in dealing with the problems of supporting operational aircraft having a large OFP component is the Sacramento ALC. It has supported the F-111F, F-111D, and FB-111A software since 1973, and its experience has supplied valuable data to other ALCs now planning for future OFP support.

The experience at Sacramento has shown that it takes more than a year to perform all the change activities from initial requirements review to the documentation and release of a new OFP flight tape. Software changes are therefore best accumulated and integrated into the baseline as a collection of changes that are processed concurrently. For all F-111s the period for this "block change" activity is 18 months long and usually involves 30 to 40 functional changes to the OFP software. However, superimposed on the block change concept is a quick reaction procedure that can be used to deal expeditiously with emergency requirements. A block diagram of the activities during the 18-month change cycle is shown in Fig. 5. This change cycle is a complete development cycle with milestones for requirements analysis/definition, establishing the design, performing the development and testing, and final implementation and documentation.

Between responsibility transfer and mid-1979, 282 changes were made to the OFPs on FB-111As, F-111Ds, and F-111Fs, for the following reasons:



Source: Sacramento ALC

Fig. 5—F-111 OFP change cycle

Change	Percent
Deficiency correction	40
Enhancements	27
New capabilities	19
Capability deletions	10
Optimization	4

Each year from 2 to 5 percent of the total code has been changed. Enough resources are available to complete all essential changes; however, manpower resources usually limit the total change activity.[7]

Figure 6 shows the change distribution over time for the F-111. Although deficiency corrections do decrease slightly over time, the decrease is less than one would initially expect, partly because new errors are inadvertently created in the process of designing and integrating new capabilities and enhancements and partly because some residual errors from the original program remain.

[7] For example, on average about one in nine change candidates have been made each change cycle. Some changes may be carried forward to the next cycle.

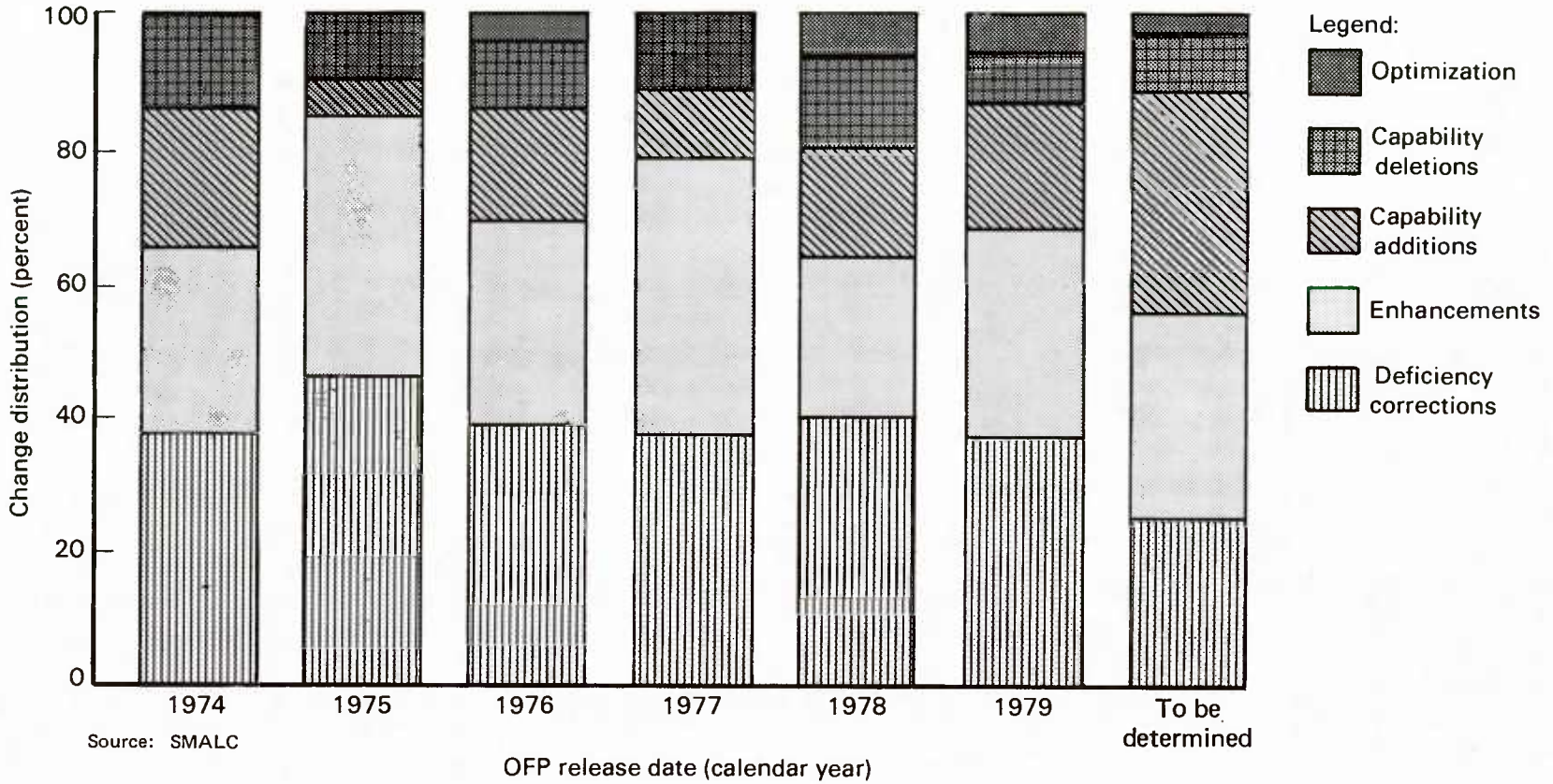


Fig. 6—F-111 OFP change history since PMRT

Different kinds of skills are used in different phases of the change cycle, and this can present problems in keeping the different professionals busy throughout the cycle. For example, the people doing validation and verification during the latter part of the cycle must be different people from those who do the design in the earlier part of the cycle. This problem is handled nicely in the F-111 program because there are three mission designation series to support. The change cycles for each mission series are staggered to begin block change cycles at six-month intervals, providing a more continuous workload for skilled personnel.

Originally, the planning for the F-16 and the F-4 at Ogden ALC was for a 12-month OFP change cycle, and there were no definite plans for cross-utilization of people between the F-4 and F-16 programs. This planning was changed to an 18-month change cycle, and multiple versions of the ARN-101 OFP on the F-4E, RF-4C, and F-4G will allow staggering of change cycles on that system. Ogden's plans also call for a cross-utilization of F-4 and F-16 mechanization engineers. For example, the F-16 navigation function engineer would participate in the decision walk-through of the F-4 navigation engineer's mechanization of the F-4 OFP.

OFP RESOURCE REQUIREMENTS

The principal resources needed for OFP support are people; specialized equipment such as support computers, system and environmental simulators; and the physical facilities such as buildings, including electricity and plumbing. Our survey has concentrated primarily on the number and kinds of people needed, and the specialized kinds of equipment required to support OFP. No attempt was made to identify or evaluate the required physical facilities.

In their planning for support of OFPs, the Air Logistics Centers do not expect to need many professional programmers. Although programming skills are required by many of their technical support people, these skills can be acquired through on-the-job training programs. The critical discipline required is engineering, primarily electrical engineering. More engineers are needed who can understand both the overall operation of a weapon system and the various component interactions. The specific job of programming or reprogramming is a small part of the OFP support function. For example, for one problem with an F-111 OFP, it took four months to isolate and design a solution and only 20 minutes (two statement changes) to carry it out. Another change required 12 people working six months on a ballistic problem that ultimately changed only 20 lines of code. Although some knowledge of programming was needed for designing a solution to these problems, the predominant skill required was engineering.

A breakdown of Oklahoma City ALC's stated skill requirements for OFP support is shown in Table 3.[8] Note that 64 percent of needed personnel are engineers, and only 1 percent are computer scientists. Some studies have shown different mixes of personnel with as high as 57 percent computer scientists and only 27 percent engineers.[9] However, the consensus at all ALCs was consistent with the Oklahoma City planning estimate shown in Table 3.[10] Both Warner Robins and San Antonio Air

[8] This skill mix also holds for the requirement analysis and design phase for ATE unit under test software support.

[9] Ref. 14.

[10] The civil service career progression path is more attractive for engineers than for programmers, hence ALCs find it easier to recruit those in engineering disciplines. This may be an additional reason ALCs prefer using engineers for software support.

Table 3

SKILL REQUIREMENTS FOR ECS SOFTWARE SUPPORT

Director of materiel management OFP & ATE support function		Test software support function
Discipline	Fraction of total(%)	
Engineer (primarily EE)	64	● Less emphasis on degreed engineering skills - WR-ALC/MAIT ~42% Engrs - SA-ALC/MATT ~27% Engrs
Electronic technician	8	
Mathematician	4	
Logistics mgt. specialist	3	
Computer system analyst	<1	
Computer scientist	<1	
Machine operator	<1	
Administrative specialist	15	● Technician skill valuable for ITA and software tasks
Steno	5	
	100%	

Source: Oklahoma City ALC

Logistics Centers indicated less need for engineering skills and greater need for technician skills.

Information about future personnel requirements in Table 4 was obtained from estimates the ALCs made while they were determining whether the support function in each new system should be performed by the government (organic support), by contract to private commercial sources, or by a combination of government and contractor resources. The approach to estimating manpower requirements varied somewhat from ALC to ALC and was based on their organizational structure and past experience. However, the basic elements of the Generic Logistics Decision Tree (as implemented by AFLC Regulation 400-03) were used as a common policy and procedures guideline.[11]

[11] AFLC Regulation 400-03 is a draft regulation to provide the policies and procedures for determining whether AFLC's commercial and industrial activities should be operated and managed by the government or a private commercial source. It implements OMB Circular A-76, DoD Directive 4100.15, and AFM 26-1.

Table 4

EXAMPLES OF RESOURCES FOR OFP SUPPORT

Personnel		
ALC	Current (Gov't/Contr.)	1986
Sacramento (MMEC, MAIT, ACD)[a]	20/55	-
Ogden (MMEC, MMET, ACDC)[a]	83/50	188
Warner Robins (MMEC, ACD)[a]	74/36	250
Oklahoma City (MM)[a]	65-75	245
Avionics integration support facilities		
~$5-15 million per system supported		

[a] These designations are office symbols and not acronyms. Symbols beginning with AC are associated with the comptroller, MM symbols are associated with materiel management and MA symbols are associated with the maintenance organizations.

Table 4 summarizes the OFP personnel resources information that we were successful in assembling. The first column shows the organization involved in OFP support at each ALC. The difference in organizational structure at each ALC is reflected in this column. The second column shows the number of people currently performing the support function. The data distinguish between government employees (military and civil service) and contractor personnel. Sacramento has 55 contractor people supporting OFP programs, and this is expected to continue. The 36 contractor personnel at Warner Robins are involved primarily in nonrecurring tasks--e.g., the setup and final checkout of the F-15 Avionics Integration Support Facility and engineering data system. The Current column represents people actually on board and is less than the

number of people authorized. Subsequent to our survey, Sacramento began hiring new personnel for the F-111 Mark II series computer replacement program and to satisfy A-10 software support responsibilities. At the time of our visit, no personnel increase was expected for the San Antonio ALC.[12]

Several ALCs will indeed require substantial additional personnel. The task of attracting, training, and retraining that many new people will present a considerable challenge. However, personnel requirements do not approach the exponential growth some have inferred from the trend depicted in Fig. 1.

COMPARISON OF ACQUISITION AND SUPPORT COSTS

Another objective of the study was to compare OFP acquisition costs and their support costs. For each individual program, costs for personnel, equipment, and contract support roughly determine support costs. Determining the acquisition costs for OFPs proved to be more difficult. Present accounting and data collection procedures do not provide the needed data. The SPOs we visited, for example, did not routinely require contractors to report software development costs separately from other costs. There are a number of rules of thumb used to estimate software development costs, but the basis for these is largely intuitive and speculative.

A report by Boeing Computer Services estimated that 40 man months are required for every 1000 statements developed.[13] The estimate includes the preparation of draft commercial documentation but not the

---

[12] Assigning the new C-X transport to San Antonio might change the personnel requirements picture by the late 1980s.

[13] Ref. 15. A statement was defined to be a fully checked out, tested, and documented computer instruction code.

detailed formal documentation required by Military Standard 483. This estimate reflects manpower costs only and does not include any computer costs. Using an FY80 industrial standard of $80,000 per man year, the Boeing estimates translate to $225 per statement. Another estimate was developed from F-16 Engineering Change Proposal data. On a large OFP change that was exclusively software the contract averaged out to $1000 per statement. Even though the change dealt exclusively with software, some level of effort was probably allocated to other aspects of the change process, so the $1000 figure is probably somewhat high. With this kind of information we can conclude only that acquisition costs are somewhere between $225 and $1000 per finished statement. The cost per instruction is much higher for complex real-time software than it is for simple software without real-time constraints. For simplicity, however, we use the average figure of $600 per program statement to come up with an acquisition cost for the F-16 OFP of approximately $28.2 million.[14]

To establish an approximate ten-year cost for support of the F-16 OFP the analysis of the Battelle Columbus Labs[15] was again used to establish personnel costs. The Battelle analysis used an OSD document[16] to establish support personnel costs. These costs are $3406 per GS-12 level man-month in FY 1980 dollars. Because some of the people are at a lower GS level, use of the $3406 per man-month figure will bias the support cost upward. The planning factor is for about 39 people (for all categories) to support the F-16 OFP in its currently

---

[14] This includes software that is planned to be supported organically--the NAV-Weapons Delivery (25K words), stores management (15K equivalent 16-bit words), and the Heads up display (7K words) programs.

[15] Ref. 16.

[16] Ref. 17.

approved configuration. Thirty-nine people for ten years at $3406 per man-month gives a ten-year personnel cost for support of the F-16 OFP of about $15.9M. The F-16 Avionics Integration Support Facility acquisition costs are $12.2 million, bringing the estimate's total ten-year OFP support cost to $28.1 million.[17] This makes the ten-year support costs about 50 percent of software life cycle costs, which is considerably less than we would expect if the trend depicted in Fig. 3 was accurate. Discounting would diminish the apparent cost of support relative to acquisition. A more abstract argument that calls Fig. 3 into question is that on average, 2 to 5 percent of OFP code typically gets changed per year, or 20 to 50 percent in ten years. If development and support costs are proportional to the amount of code manipulated, then a 3:1 ratio of support to development costs will probably not apply.[18]

Admittedly these numbers are quite rough. However, they represent the best that can be done with the available data. It is important to recognize how imprecise this information is and how useful better information would be to the planning process.

EFFECT OF HIGHER ORDER LANGUAGES ON OFP SUPPORT

The growth in ECS software support requirements has prompted a search for means to enhance the productivity of the individual programmer. The use of higher order languages instead of assembly

---

[17] The Avionics Integration Support Facility is used for more than just OFP support, so charging its full cost against OFP support tends to overstate support costs somewhat.

[18] Supporting a computer program that saturates all available memory, as is often the case with OFPs, increases the support effort per statement, driving support costs closer to parity with development costs.

languages is one approach to achieve substantial reductions in the level of effort required to support OFPs. To further evaluate this hypothesis we requested the opinions of responsible individuals in ALCs that are or will be supporting OFPs. A consistent opinion from managers responsible for OFP support was that although HOLs offer a number of advantages (mostly in the nonrecurring cost area), they would not greatly reduce the time to effect a change or the costs of support.

To move beyond the level of considered opinion in this area we merged quantitative information about the resources required for recurring F-111 OFP support obtained from the Sacramento ALC with an interpretation of a Battelle Columbus Laboratory comparison of various software development alternatives for F-111A and F-111E digital-bomb-navigation system operational flight programs.[19] The Battelle study measured savings possible in recurring OFP support through the use of HOL (JOVIAL J73/I) instead of assembly language. Their investigation indicated that use of the HOL, coupled with structured programming techniques, primarily influenced the development, testing, and documentation activities and yielded savings in each activity of from 20 to 30 percent.

We considered these savings in the context of all the activities that contribute to the release of an F-111 OFP. A typical distribution of the recurring level of effort to support an F-111 OFP is depicted in Fig. 7. Projected savings due to the use of an HOL and structured programming techniques are shaded. HOLs offer impressive savings in a few of the individual activities, but when considered in the context of all the activities that contribute to the release of an OFP, savings are on the order of 10 percent.

---

[19] Ref. 16.

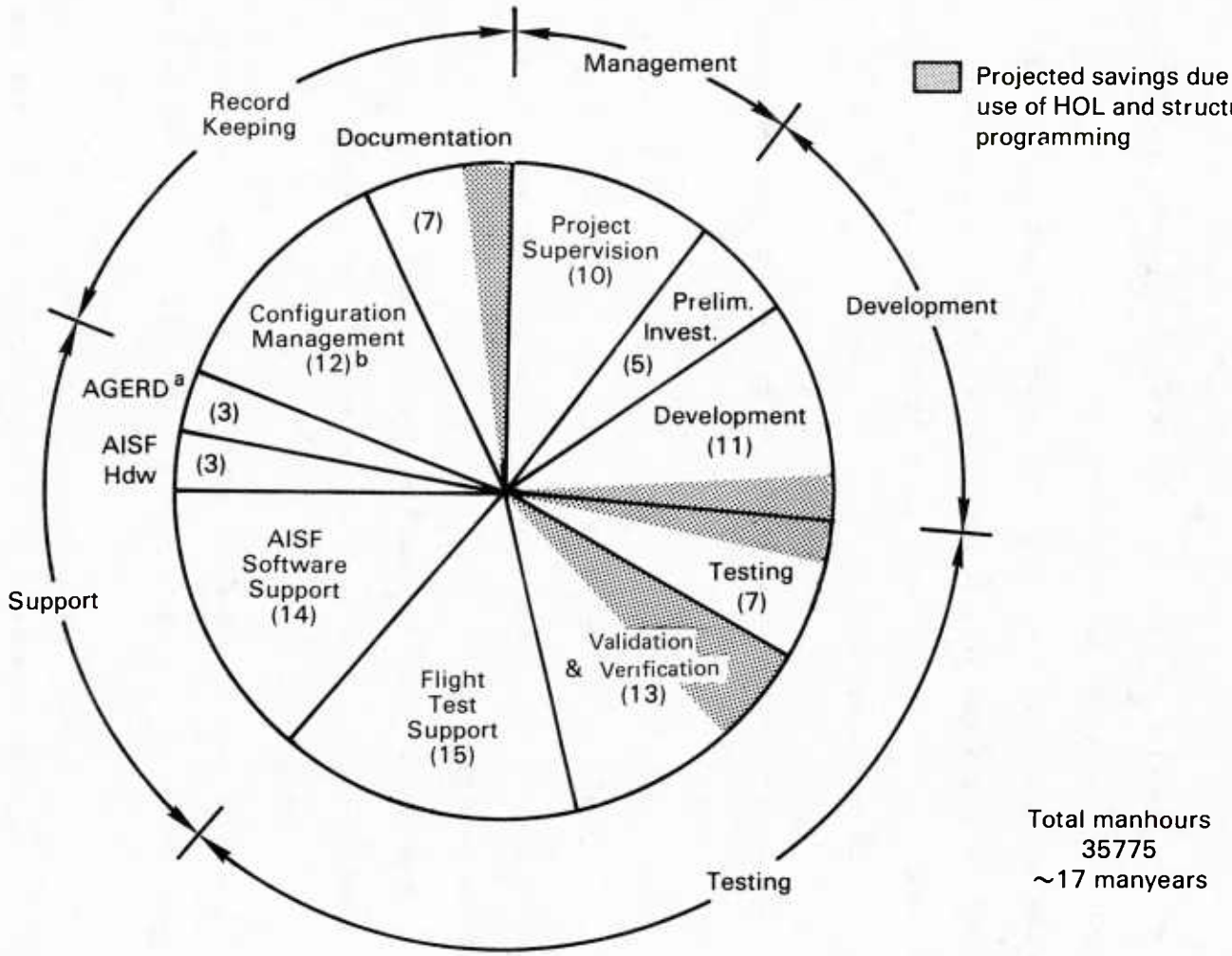


Fig. 7 — Potential effect of an HOL on recurring OFP support effort

[a] Aerospace ground equipment requirements documentation.
[b] Numbers in parentheses denote fraction of total support activity devoted to each task. (e.g. configuration management accounts for 12% of the 35775 man hours to support

When we queried ALC personnel about whether the use of HOLs might greatly compress the calendar time required to release an OFP, their expectation was that only modest savings were probable, perhaps on the order of 5 percent. In comparing support activities depicted earlier in Fig. 5 with the fraction of those activities materially affected by the use of a HOL, we come to a similar conclusion.

The foregoing is not intended as an argument against the use of higher order languages; rather, it attempts to put the savings they offer for recurring OFP support into perspective. HOLs have several positive aspects that must be considered.

1. Programs are usually easier to read for someone not familiar with the code.
2. It is generally easier to find personnel trained in HOL usage, or to train new personnel to use HOLs.
3. HOLs are usually less machine dependent and thus more amenable to standardization.
4. Statements (1) and (2) above imply that HOLs have better native documentation--a feeling shared by the Air Force and contractors alike.
5. Statements (1), (2), and (4) imply that HOLs ease the change process and not just at the programming step, but during design too.

Despite such clear advantages, most of the benefits of HOL use are obtained in combination with a number of other well-recognized software engineering practices; examples are structured design and structured

programming techniques, careful technical reviews, and strict configuration management from a firm base line.

By itself, the use of a HOL cannot be expected to produce a major reduction in OFP support expenditures or in the time required to produce changes.

III. THE AUTOMATIC TEST EQUIPMENT DEVELOPMENT AND SUPPORT PROCESS

To gain a better understanding of the automatic test equipment (ATE) software development and support process, we have identified some of the key organizational interactions and characterized certain current and expected support responsibilities and workloads. We identifed now-visible resource needs and personnel skill requirements and considered the status of the language standardization issue. In addition, we used available data to compare acquisition and support costs for ATE software and hardware.

DEVELOPMENT OF ATE SOFTWARE

Typically, the acquisition of ATE and its associated software is the responsibility of the System Program Office, although other Air Force agencies may be involved. The SPO organization is the one that must commit contractual responsibility, usually to the prime contractor, for the development of the Test Requirements Documents (TRDs), which can be expensive. The TRDs required to support a representative system may cost anywhere from $7 to $25 million. They contain detailed specifications for what must be tested and the sequence of testing for each individual system component at each maintenance level. TRDs are also important because they are the basis for the design of depot and intermediate level ATE.

After delivery of the TRDs, the SPO must go through an evaluation process to select development contractors for the ATE equipment and software. This evaluation is complex because, in addition to the testing requirements specified in the TRDs, it must take into

consideration ATE already in the Air Force inventory, total life cycle costs of various alternatives, and the effect of each alternative on the supporting agency.

A number of Air Force organizations are involved formally or informally in the ATE acquisition and support process. In Fig. 8 some of the lines of interaction between organizational boxes have been omitted for clarity; however, Fig. 8 represents the principal organizations and major interfaces involved.

Before discussing the roles played by each organization, we define the term "ATE software" more clearly. There are three major kinds of software involved in ATE--executive, unit under test (UUT), and support software. This discussion examines the first two.

Executive software, more often referred to as operating system or supervisory software, creates and maintains the environment within which UUT software can be used. Executive software functions include scheduling the sequence of execution for individual UUT programs, allocating memory resources to each scheduled program, monitoring input and output requirements, handling necessary interruptions to the job stream, managing requests for standard library routines (e.g., for computing mathematical functions), and the like. Once developed, executive software is not changed very often, and then seldom by the Air Force. Improvement of executive software and the elimination of bugs are functions typically carried out by the original vendor, who tries to consider the effect of changes on all the users of its test equipment.[1] Such changes are distributed periodically to all users of

[1] The Air Force is actually a small user of ATE compared with commercial industry. Capitalizing on commercial ATE development is a cost-effective approach for the Air Force to take.

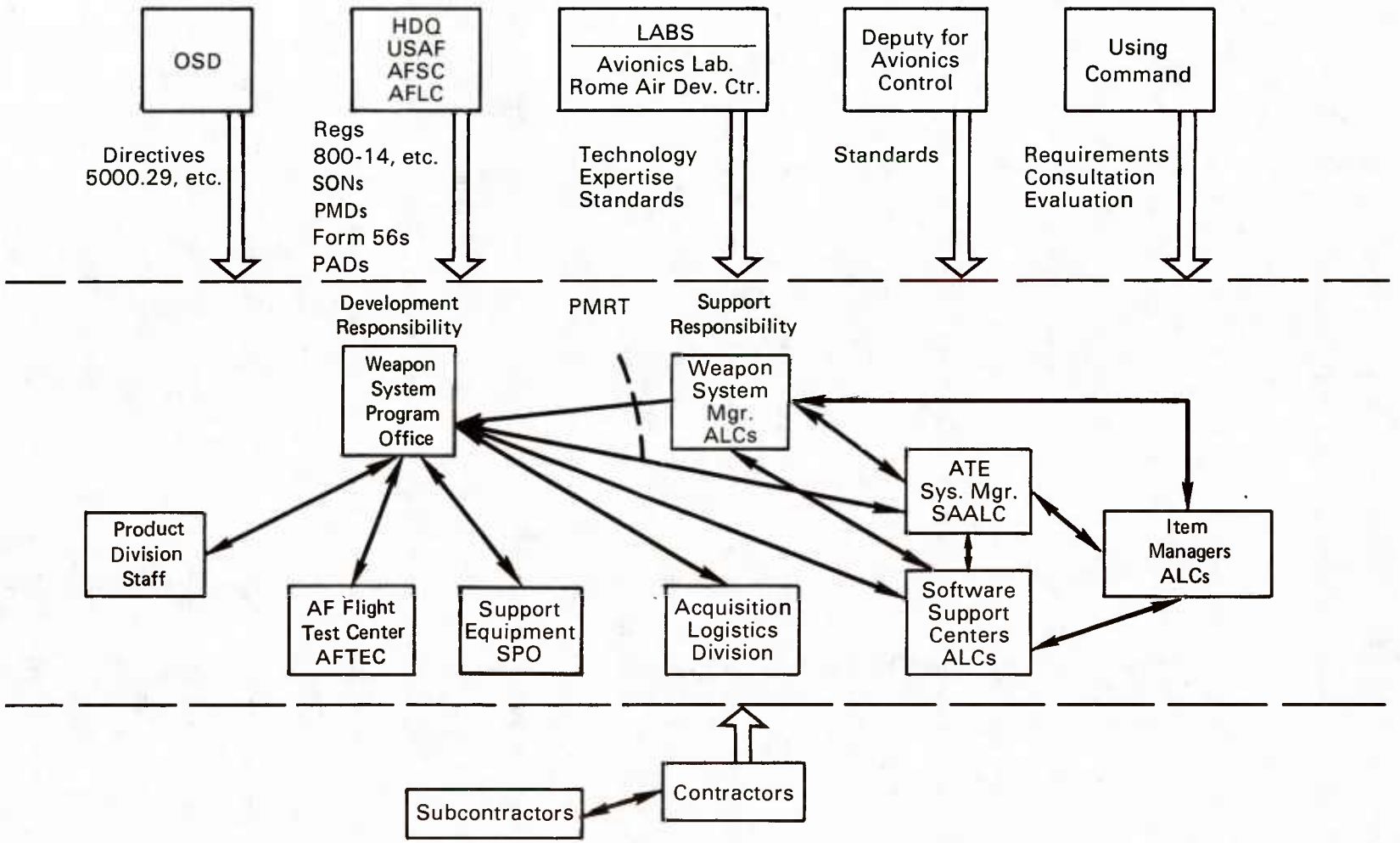


Fig. 8 — Major organizations involved in the acquisition and support of ATE software

the equipment including the Air Force. An example is the way in which IBM supports the operating systems for their computers. Executive software can be very large and complex--sometimes numbering in the millions of words--another reason for not changing it indiscriminately.[2]

The executive software and its standard supporting service is typically purchased by the SPO as off-the-shelf items along with the ATE hardware. Once acquired, system responsibility for ATE resides at the San Antonio Air Logistics Center, as depicted in Fig. 8.

In addition to the executive software, computer programs are developed to test each component of a given system. There is usually a different program for each unit of equipment being tested and it is referred to as UUT software. At the base level (Operating Commands) the units under test are line replaceable units, and the test equipment strives to locate faults down to the printed circuit board level. At the depot, the test equipment is expected to isolate faults down to the individual component level--e.g., a resistor, a chip, a capacitor. For a fighter aircraft there would typically be 100 to 150 UUT programs at the intermediate shop (base level), whereas at the depot level there might be 800 to 1500 UUT programs.

Along with a unique UUT program for each item being tested, there is also a unique piece of hardware called an interface test adapter, which mates one or more UUTs electrically and mechanically to the test

---

[2] Trying to generalize about executive software is sometimes misleading. Some test equipment works in conjunction with large mainframe computers that may involve hundreds of thousands of lines of code, whereas other test equipment has totally self-contained processors using only a few thousand lines of software.

equipment. In some newer systems, the adapter itself has been made programmable to increase its flexibility. This adds to the support burden, however.

Thus, the SPO must contract not only for the ATE hardware and executive software, but also for UUT software and interface test adapters for each item tested at intermediate and depot levels.

Organizations available to the SPO for formal or informal assistance in the acquisition of ATE software are shown in Fig. 8. The charter of the Acquisition Logistics Division (ALD) calls for it to assist the SPO in matters relative to logistics support of Air Force systems. However, as mentioned in the discussion of Operational Flight Programs, at the time of our survey (1979-80) ALD had only nine people with appreciable software expertise, and they were spread over the software areas of many Air Force programs. As a consequence, ALD cannot always lend a great deal of assistance on ATE software acquisition questions.

The Support Equipment (SE) SPO is another agency available to assist the system SPO. The SE SPO, jointly manned by AFSC and AFLC, was formed to improve the management of support equipment acquisition. As of early 1980 there were a total of 67 people in the SE SPO, with 13 collocated at different weapon systems program offices to assist in support equipment acquisition. However, at the time of our interview, only four or five of their people were "software conversant," and they were assigned to the Modular Automatic Test Equipment development program and therefore not available to assist the weapon system SPOs in ATE software acquisition. The current net result is that the SE SPO cannot provide major assistance to SPOs in ATE software acquisition.

Other Air Force agencies available to the SPO for assistance in ATE software acquisition are the Maintenance Divisions, including the Software Support Centers, and the Materiel Management Divisions of the different ALCs. These organizations have a fairly high level of expertise in ATE software because they are in the business of developing and supporting UUT programs. Their assistance to the SPO in the design and production of UUT software as well as for the design and fabrication of the associated interface test adapters has at times resulted in savings compared with commercial acquisition and has had the added benefit of preparing the ALCs for their eventual role in supporting the UUT software and interface test adapters. The ALCs can also assist the SPO by helping both to prepare contracts for commercial acquisition of UUT software and by monitoring the progress of these contracts. This same function can be supplied to the SPO by a contractor under a technical assistance contract.

Other ATE-relevant Air Force organizations are the Operating Commands and the Air Force Test and Evaluation Center (AFTEC). These organizations may participate in the selection process for ATE, but they are responsible for evaluating the operational effectiveness of both the system and its support equipment.

SUPPORT OF ATE SOFTWARE

After ATE has been acquired and accepted by the SPO and its supporting organizations, it is transferred to the using organization. For instance, intermediate-level ATE is transferred to the Operating Command, and depot-level ATE is transferred to an ALC. Overall system responsibility for ATE is transferred to the Air Logistics Center at San

Antonio. Even though intermediate-level ATE resides with the Operating Command, the support of that ATE is done by the ALC that has item responsibility.

Different categories of ATE software undergo changes at different rates. For example, on an annual basis less than 1 percent of ATE executive control software may be changed after program management responsibility transfer. By comparison the figures for support[3] and UUT software are 1 to 5 percent and 10 to 20 percent respectively.

The 10 to 20 percent figure for UUT software represents a considerable organic support burden and frequently includes hardware modifications to the relevant interface test adapters. In some cases, UUT software changes necessitate completely new adapters. Examples of the number of separate UUT software programs currently supported and expected for the future are shown in Table 5.[4]

ATE software changes are not accumulated and handled in blocks as with OFP software. Instead, each change is handled on an individual basis to minimize the time from the recognition of a deficiency to its correction. This is an acceptable approach because each UUT usually requires unique test software. The time to process individual changes to UUT software can range from two weeks to 13 months. Figure 9 depicts the different steps in the UUT software change process. The division of responsibility in the change process between Materiel Management and

---

[3] The support software are tools used by the software support centers in developing and supporting the UUT software. They are supported principally by those organizations.

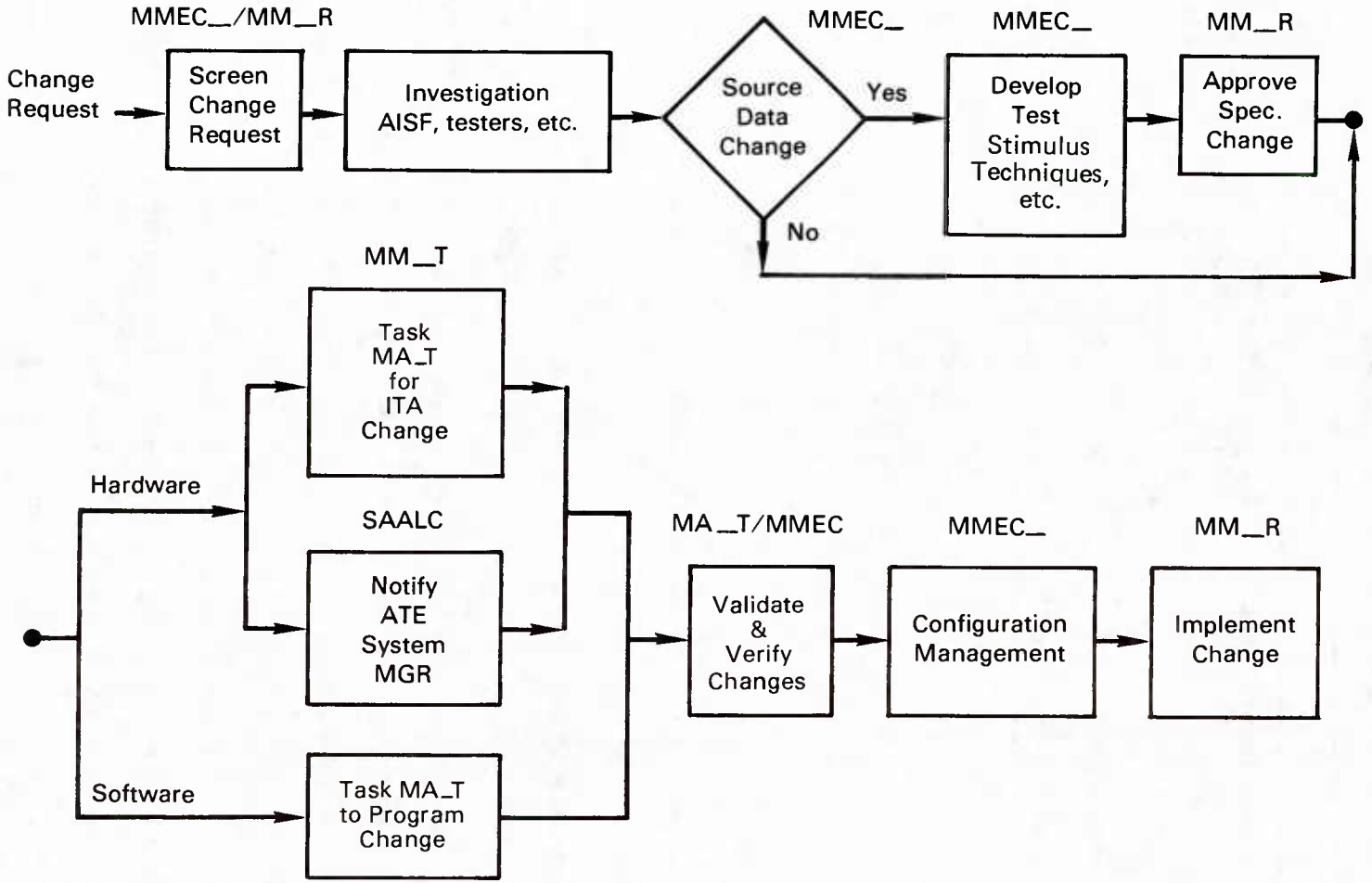
[4] There may be a small amount of double counting in the totals of Table 5. For example, the material management organization may do problem review and solution identification and then task one of the software support centers to make the change. Each counts the program in their totals.

Table 5
EXAMPLES OF CURRENT AND FUTURE NUMBER OF
UUT SOFTWARE PROGRAMS SUPPORTED

ALC	Current	Future
Warner Robins (MMEC)	2000	4000-6000 (FY83)
(MAIT)	1800	3700 (FY82)
Ogden (MACT)	2400	3300+ (FY83)
Sacramento (MMEC)	1700	5000+ (FY85)
Oklahoma City (MMEC)	1500	N/A

Maintenance organizations varies at the individual ALCs, with the extreme being the software support center at the Ogden ALC, which processes requests from start to finish within its organization. Changes are initiated either because of a deficiency in an existing test program or because there is a requirement for a new capability. If the change is to correct a deficiency, it is initiated either by a field organization complaint or by the Maintenance Organization of the ALC. Changes initiated by the Materiel Management branch of the ALC are usually because of a requirement for a new test capability.

The appropriate item manager for the UUT and engineers from the materiel management organization screen the change requests for validity. This may involve more than a desk-top review and include



[a] Except at OOALC

Source: WRALC

Fig. 9—Test software change process[a]

consideration of the potential effect on test equipment or avionics integration support facility. If the early screening indicates that the change is reasonable and sensible, a materiel improvement project is initiated. If the change involves a source data change[5] then engineering actions may be required to develop new techniques for testing the altered circuit board or line replaceable unit. After approval of the Test Specification change, one of the software support centers or a private contractor is tasked to make the software and hardware changes to the interface test adapter, if required. The ATE system manager at San Antonio ALC must also be notified of the changes. The final steps before implementation are to validate and verify the changes and to prepare the documentation.

Typical timing for this process would include roughly two weeks for screening, investigation, and approval of specification changes, from one to five months at the software support center, and from three to six months to update the documentation and issue the Time Compliance Technical Order. Thus, a typical program change might take from six to twelve months. One reason for this long change cycle is that once the software support center starts implementing the change, they may find that the problem is not what it was originally thought to be. This requires them to broaden the scope of their inquiry to accomplish the originally intended change in capability.

It may be that the test equipment is simply not detecting bad UUTs. If the repair line is unable to turn out functional units, then

---

[5] A source data change is a change in the fundamental layout of an item--e.g., a change in an engineering drawing that defines a printed circuit board.

emergency changes are required for the ATE. To expedite these urgent changes the ALCs can use a "production stoppage supplement," which allows them to issue a new test program without an accompanying time compliance technical order. Because the production stoppage supplement is active for only a limited time, follow-up actions must be taken to effect the change in a complete materiel improvement project cycle to make the change permanent.[6]

HIGHER ORDER LANGUAGES FOR ATE

The higher order language issue is quite different for the ATE software arena than for OFPs. For one thing, memory space is a less critical limitation in the hardware environment; consequently, there is less motivation to use assembly language as a means of saving storage. Nor is there a large real-time computing requirement, so that fine tuning programs for real-time operation are not usually required. Tuning is required for increasing the speed of test programs to reduce the time for testing an individual UUT.

The Department of Defense has required that ATLAS be used as the standard language for ATE software. Most commercial firms already use some version of ATLAS, and the Air Force's modular automatic test equipment program is committed to it as well. If an ATE developer wants to use a language other than ATLAS, it must ask AFSC for a waiver. None have been granted to date. On the surface then, language standardization is not a major problem in this particular ECS area. However, the support equipment SPO has counted 73 different versions of the ATLAS language currently in use. Warner Robins ALC has identified

---

[6] Some ALC personnel have suggested that one way to reduce the change process burden might be to accumulate these changes and process them in blocks.

42 different languages (primarily different versions of ATLAS) that they are supporting in ATE. This problem is well known throughout the Air Force ATE community, and language standardization has been included as a goal for the modular automatic test equipment program.

ATE RESOURCE REQUIREMENTS

The ALCs do not expect that ATE software support will drive a large future demand for professional programmers. The principal skill categories needed are those of technician and engineer. These people must first have a solid background in the testing business, and they must have a functional understanding of the UUTs for which they are designing test programs.

Given this basis, the requisite programming skills can be acquired through their training program. Although the degreed engineering skills are extremely important, they are present in the ATE software support in lower proportions than in the OFP business. As was indicated earlier, more than half of the personnel in OFP support were degreed engineers, whereas in ATE software support, there are fewer; e.g., about 42 percent of the Warner Robins ALC/MAIT personnel are engineers, and at Sacramento ALC/MAIT approximately 27 percent are engineers. Most software support centers rely heavily on technicians for programming software changes, and in designing and building interface test adapters.

For the Air Force organizations polled, Table 6 shows the number of Air Force people involved in ATE program development and support for late 1979 and early 1980. Also shown are the expected future personnel requirements; the right-hand column shows the approximate annual percentage increase expected. The expectations of software support

Table 6

EXAMPLES OF PERSONNEL FOR ATE SOFTWARE SUPPORT

ALC	Current			Future
	Gov't/Cont.	Dev.	Support software	
Warner Robins (MAIT)	110	40		251 (FY82)
(MMEC)	28			60 (FY83)
Ogden (MACT)	70/2	24		100 (FY83)
Sacramento (MMEC/ACD)	17/6			46-92 (FY83)
(MAIT)	134	90	11	
Oklahoma City (MM)	10-15			36 (FY83)
(MATT)	46		33	
San Antonio (MATT)	111		20	

center branch chiefs about personnel growth vary considerably from one ALC to another.

Despite their name, software support centers do more than just support software of developed systems. Considerable effort is devoted to developing new unit under test software for the different product divisions of AFSC (Aeronautical Systems Division, Electronics Systems Division and Space Division). For example, 30 to 40 percent of the effort at Warner Robins ALC/MAIT is devoted to developing new unit under test software, much of it for AFSC. At Sacramento ALC/MAIT, a similar situation exists--about two-thirds of their people do software development work for ESD. Likewise at Ogden ALC/MACT, about a third of their people develop new UUT software. Personnel move back and forth

between the development of new software and the support of existing software in order to stay abreast of new test software development techniques.

Personnel at software support centers also perform tasks that fall outside the categories of developing or supporting software. For example, at Ogden ALC/MACT, the distribution of effort is estimated to be 60 percent for analysis, design, and programming, 30 percent for interface test adapter design and fabrication (an intermediate device that allows the UUT to communicate with the test equipment), and 10 percent for miscellaneous tasks such as drafting and documentation.

Our assessment indicates the ALCs will experience a substantial increase in the number of people needed early in this decade to support ATE software, and that the increase will be on the order of two to three hundred. The ALCs are convinced that engineers and technicians are needed to accomplish the support task rather than professional programmers.

ACQUISITION AND SUPPORT COSTS FOR ATE HARDWARE AND SOFTWARE

One objective of the study was to examine the validity of the perception that embedded computer systems software is consuming an ever-increasing part of the total acquisition dollar, and that software would eventually consume about 80 percent of system life cycle costs (see Fig. 3). Some information was available to examine this perception for ATE; however, for many systems it was not possible to break out software costs. The software development costs are usually embedded in the station development/delivery costs. The software support costs are divided between engineering and configuration management, with the engineering time embedded in a fixed price sustaining engineering

contract line item. Although the contractor develops a software cost estimate to establish a total price, corporate policies usually forbid the release of such information.

In response to a Hq AFSC initiative the F-16 SPO assembled a software cost break out. Some of the elements are necessarily estimates because, as is usually the case, their contracts do not specifically require software cost reporting. This F-16 information, along with some information published by the ATE system managers at San Antonio ALC,[7] enabled the preliminary comparison shown in Table 7. The comparison of F-16 ATE hardware and software acquisition costs does not support the life cycle cost distribution depicted in Fig. 3 and found in numerous discussions of the subject.

Several examples were available to compare ATE software acquisition and support costs. The F-16 software support costs estimates were given a plus or minus 25 percent confidence factor by the SPO. A plus or minus 10 percent confidence in the acquisition costs gave a spread in the comparison. Table 7 shows that ATE software acquisition costs exceed software support costs by a factor of 1.4 to 2.9. Three different systems considered by the ATE system manager[8] gave results similar to those indicated in Table 7. If constant dollar cost ratios are used, then the dominance of acquisition costs is even more striking. Discounting the out-year support costs would reinforce the result.

---

[7] Ref. 18.
[8] Ref. 18.

Table 7
ATE Acquisition and Support Costs

Hardware vs. software acquisition costs

	F-16 intermediate & depot level ATE Costs (millions then yr. $)		
	Hardware	Software	Ratio (HDW/SW costs)
Development	101	59	1.7
Procurement	143	8	18
(±10%)	220-268	60-74	3-4.5

Software acquisition vs. support costs — 4 examples

	Then yr $ cost ratio (acq./10 yr. support)	Constant $ cost ratio (acq./10 yr. support)
F-16 int. & depot level software	1.4-2.9	N/A
Commercial automated tester with		
- Reprogrammable firmware	2.2	2.5
- Minicomputer	2.6	3.1
Mil spec ATE	3.0	N/A

ATE SOFTWARE SUMMARY

Although all the ALCs expect to experience personnel growth in the next five years, this increase will be measured in the low hundreds at the particular ALCs that are undertaking new responsibilities. The ALCs believe technicians and engineers will be needed more than professional programmers.

Because memory space limitations are not as critical in automatic test equipment hardware as in operational flight programs, and DoD has established ATLAS as the standard higher order language to be used in ATE, the higher order versus assembly language issue is not as relevant to ATE. There is, however, still a need to reduce or at least slow the growth in the number of ATLAS dialects.

Although empirical cost information on ATE software acquisition and support is limited, what is available does not support the contention that software support costs are emerging as the dominant element of embedded computer system life-cycle costs.

IV. DEVELOPMENT AND SUPPORT OF SOFTWARE FOR AIRCREW TRAINING DEVICES

INTRODUCTION

Dramatic escalations in the costs of jet fuel and flight hardware and the increased fidelity of simulator systems made possible by recent technological advances have encouraged the Air Force to rely more and more on aircrew training devices (ATDs). These devices can range from highly sophisticated weapon system trainers having extensive visual displays, full motion bases, and representations of the radar and land mass, to operational flight trainers, to simpler cockpit procedures trainers. The computer resource hardware in these systems is primarily commercial off-the-shelf computers and peripherals. Most of the newer devices use software in ever greater quantities; hence ATDs represent an important and growing embedded computer software support responsibility for the Air Force.[1] ATD-related software generally falls into one of three broad categories: (1) the system software used to operate the training device, (2) the software used to support the system software, and (3) ATE software used to test the training device.

ORGANIZATIONAL FRAMEWORK

Individual weapon system SPOs and the Deputy for Simulators (Simulator SPO) are the principal organizations having management responsibility for ATD acquisition (see Fig. 10). In the past, weapon system SPOs traditionally managed the acquisition of training devices as

---

[1] Air Force Regulation 50-11, "Management and Utilization of Training Devices," defines all the various types of training devices, of which aircrew training devices are only a subset.

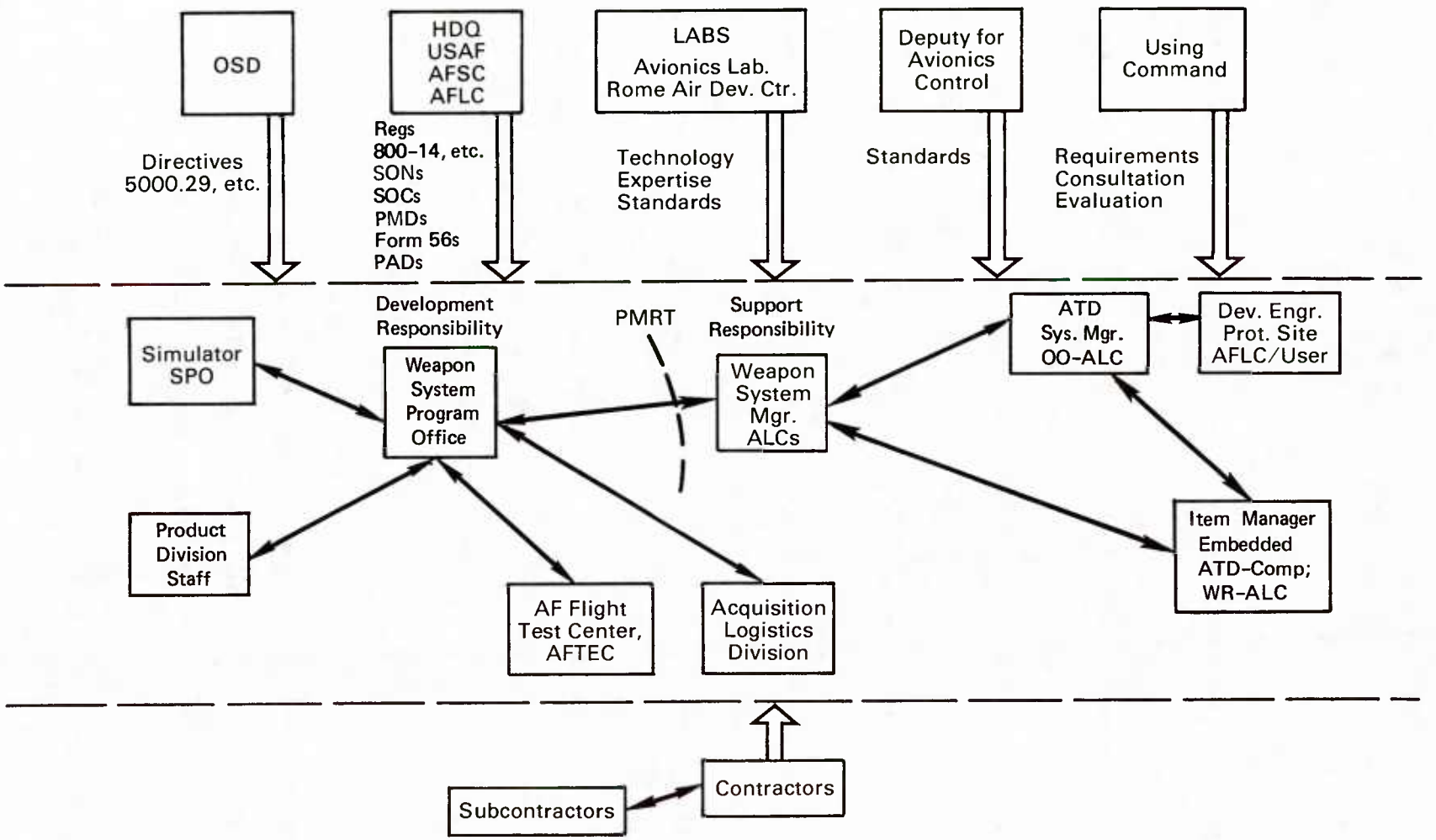


Fig. 10 – Acquisition and support of ATD software: major organizations

part of their responsibility for acquiring weapon systems. In 1974, however, the Air Force created the Deputy for Simulators as a three-letter organization to create a center of expertise for simulator acquisition. By the summer of 1979, it had achieved a two letter organizational status, putting it on an equal footing with major weapon system SPOs.

In the present Aeronautical Systems Division, the Simulator SPO occupies a much more influential position than the Support Equipment SPO, which assists in the acquisition of support equipment. The Simulator SPO is so staffed that a weapon system SPO director can delegate simulator acquisition responsibility to the SPO. Funds for simulator acquisition are program line items, and the money is allocated at the Air Staff level. As a consequence, once the funds are programmed, the weapon system SPO director does not have discretionary authority to reprogram funds away from the simulator to other areas he deems of greater importance, such as sometimes occurs with support equipment over which he has budget authority. Authority for configuration changes resides at the Simulator SPO as well.

The Simulator SPO does not currently manage all Air Force simulator acquisitions. There is a residual body of programs that began before the creation of the Simulator SPO and for which the SPO has not taken on acquisition responsibility (e.g., the F-15). In some other programs, weapon system SPO directors have elected to not use the Simulator SPO (e.g., aircraft maintenance trainers, the KC-10). In addition, AFSC (ESD and SAMSO) procures numerous training devices for electronic and missile systems. Nonetheless, the Air Force is moving in the direction

of giving the Simulator SPO acquisition responsibility for more and more systems.

The key development-support interface is between either the Simulator or the weapon system SPO and the aircrew training device system manager at Ogden (MMF) ALC, which is responsible for the executive management of ATDs. Ogden has support responsibility for trainer-peculiar hardware and software after responsibility transfer and manages the recurring hardware and software changes necessary to insure that the ATD accurately reflects the primary weapon system.

In contrast to the situation in operational flight program and automatic test equipment software, user command support teams play an active role in the support of aircrew training device software at Development Engineering Prototype Sites (DEPS), engineering facilities collocated with operational simulators that serve a function similar to that of the Avionics Integration Support Facility for operational flight programs.

The Warner Robins ALC is also involved in ATD software support and under Federal Supply Group regulations is the item manager responsible for supporting embedded ATD computers and peripherals. In contrast to Ogden, Warner Robins is responsible for supporting the software of the ECS that is not weapon specific. Needless to say, in addition to these Air Force organizations, contractors are heavily involved in the ATD software support process, particularly for major changes beyond the scope of organic support.

MEASURES OF RESPONSIBILITIES

The software support task for simulators is sizable today and shows every indication of large growth in the future. At the end of 1979, the system manager for ATDs at Ogden identified 69 separate aircrew training device items in operational status and 27 in acquisition.[2] Many of these ATDs use multiple computers for the simulation task. That same accounting suggested that the memory size of the systems in acquisition will be on average three times greater than the memory size of already operational systems.

In the spring of 1980, the Simulator SPO identified 37 simulator-related contracts with a total obligation of greater than $400 million through 1986. It further identified 13 systems with training device requirements awaiting Program Management Directives through FY 1983.

RESOURCES

To accomplish its simulator acquisition tasks, in 1980 the Simulator SPO had 112 people loaned from different directorates at ASD engineering and 89 assigned directly to the SPO for a total of 201. That total included 22 people who devoted most of their attention to activities relating to computer resources, including the position of Software Technical Advisor, created in October 1979. Simulator SPO personnel indicated that in their organization the predominant engineering skill requirement is the possession of a good understanding of digital electronics, which implies the need for an electrical engineering background. There is also, however, a continuing need for hydraulics engineers to oversee development of simulator motion systems.

[2] Most of these items have been (or will be) procured in multiple quantities.

Our survey did not extend to the measurement of individual contractor support activities before program management responsibility transfer (PMRT). Our only benchmark is the F-15 program, which is well beyond initial development and awaiting PMRT. Goodyear officials indicated they devote about three to six manyears of effort annually to engineering change proposal activities, with an additional two to three people full-time to handle operating system equipment.[3]

Our survey on the support side extended only to the ATD System Manager's organization (MMF) at the Ogden ALC. That organization has approximately 190 personnel, 80 to 90 of whom deal with computer resource issues daily. Of those, about 20 to 25 people in the Ogden organization deal almost exclusively with software-specific matters, including the review of software engineering data (e.g., Part 2 specifications). In addition, Ogden has about eight engineers stationed at the various Development Engineering Prototype Sites (DEPS) acting as ALC resident engineers.[4]

Typically each Ogden engineer is responsible for managing the software associated with four to five simulators.[5] These engineers

---

[3] Personal communication with Jay Wilcox of Goodyear Aerospace, June 1980.

[4] Personnel estimates from System Manager for ATDs, Ogden ALC.

[5] Trying to generalize about how much software this might represent is extremely difficult, given the diversity of simulator types used by the Air Force. Perhaps representative of one of the Air Force's more complex simulators, the F-15 has a software package of more than 400 software modules. Modules typically range in length from 20 to 2000 lines of code. The system has approximately 300,000 punched cards of contractually deliverable source statements to operate the simulator. Only a subset of this software would be subject to recurring change activity. Personal communication with Jim Miller, Goodyear Aerospace, June 1980.

serve an engineering management function but do not actually participate in making software changes. The field representatives do participate in making changes, but not all DEPS have field representatives.

Ogden relies heavily on contractor engineering support, about $50 million annually with an additional $4 million for field maintenance. This figure is growing. Although no formal accounting split is made between software and hardware contracts, the software support work is estimated at between $2 and $3 million.

Except for the Ogden field representatives, only user command personnel man the DEPS facilities. Although we have no totals across all locations, a former system engineer/analyst provided a breakdown of DEPS personnel at the Randolph AFB facility, which is used to train T-37 and T-38 pilots. That facility includes a user command pilot in charge, an Ogden field representative, and six user command personnel, two performing software support, one configuration accounting, and three hardware support functions. These six enlisted personnel fall into the 51 and 34 series specialty code classification.[6]

SUPPORT PROCESS

Aircrew training device software is supported using the Development Engineering Prototypes Sites concept. Each trainer model has one designated operational location featuring additional personnel from the using command, perhaps an Ogden representative, additional software tools, offline support peripheral devices, and software documentation to support the ATD software.

---

[6] Section V quantifies some of the difficulties the Air Force has had in retaining these personnel.

Suggestions for changes come from the weapon system manager's office (SPO or ALC), users, DEPS personnel, and hardware item managers. Modifications to keep simulators compatible with weapon systems constitute from 50 to 80 percent of the total change workload, and most major changes are accomplished by contractors using the engineering change proposal process, with Ogden ALC and the DEPS team participating by evaluating the adequacy of the contractor changes and testing the changes after they have been developed.

System changes that can be accomplished organically follow the "Quick Modification Process." The DEPS team performs the initial feasibility study and submits results to the user command. With approval of the concept from the user, the DEPS personnel prototype the change and submit it for approval successively to the configuration control boards of the affected commands, the ATD system manager, and the ATD configuration control board. When the changes are approved, Ogden ALC personnel develop a Time Compliance Technical Order (TCTO) to explain to field personnel how the change affects the function of the training device, update the program documentation, prepare modification kits if hardware is involved, and distribute the change to the other trainer locations.[7]

Changes to simulator software almost inevitably lag changes in the weapon system, even with the Quick-Mod process. Typically, simulator

---

[7] When only minor changes to software are involved, the ATD system manager may accumulate ten such changes or wait until a year has passed, whichever comes sooner, and then issue one formal block change that changes the ATD configuration baseline. In the interim, the other operational trainer sites rely on the DEPS for red-lined TCTOs and program tapes incorporating each change. This process reduces the time to field a change and relieves the bureaucratic burden of processing each minor change individually.

changes can lag weapon system changes by 6 to 24 months, although we learned of several contractual and technical efforts aimed at making weapon system and simulator changes more concurrent. In contrast to most simulators, which emulate the on-board computer and operational flight programs, the F-16 simulator will use the weapon system's actual embedded computer and OFP, which theoretically should make it easier to maintain consistency between the weapon system and the simulator.

In future coding efforts, the F-4G program is planning to use a structured programming approach for the OFP in an effort to provide early information to the simulator contractor before all the OFP coding is actually finished. Although the use of structured programming was primarily driven by a desire to enhance the supportability of the OFP, it is hoped that it will also result in more expeditious development of code for the OFP and the simulator.

Efforts toward greater concurrency extend to development as well. In the B-52 Offensive Avionics System (OAS) development program, Boeing has a data item requiring that it deliver critical simulator data information to the simulator contractor at the earliest possible date. This measure is intended to keep simulator development in step with OAS software development.

LANGUAGE ISSUES

A listing compiled by the ATD system manager identifies assembly language as the most frequently used language for operational ATDs. Complete tabulations are not available, although discussions with Simulator SPO personnel suggest that for new simulator software, the present trend is to a greater use of FORTRAN in preference to assembly language. Some suggested that newer simulator software typically

consists of 80 to 90 percent FORTRAN and the remainder assembly language. Simulator SPO personnel believe that hopes for considerable standardization of simulator functions at a coding level are probably not realistic, even if a standard higher order language is adopted. Although some measure of standardization may be achieved at a modeling level, the modeling of aircraft functions is not static. Users are demanding greater and greater fidelity, and there is little support for standardization of simulator functions in such a dynamic environment.

Aircrew training device development and support personnel identified several particularly vexing problems facing the simulator community today: one is the difficulty in attracting and retaining skilled personnel, including both electrical engineers and technicians, and another is adjusting to the rapid pace of computer technology, which offers the Air Force opportunities for new capabilities but complicates its hardware and software support effort.

V. PROBLEM AREAS AND EMERGING ISSUES

PROBLEM AREAS

Many of the problem areas discussed in this section have been identified in several earlier studies.[1] In this section we document the effects these problems have on software support for embedded computers. To the extent we are aware of them, we will also identify Air Force initiatives that have been directed toward easing these problems. The reader should bear in mind, however, that the present document is preliminary as it applies to problem evaluation and solution identification.

Table 8 is an unranked listing of the problem areas identified during recent interviews with personnel who are involved in software life-cycle support activities at the Air Logistics Centers, at selected ASD System Program Offices, and at other relevant organizations.

As we discuss what we have learned about each area, the frame of reference will vary according to the contexts in which the problems were encountered. Thus, some discussions will be couched in terms of operational flight programs, some in terms of automatic test equipment, and some in terms of aircrew training devices. Even so, many of the problems have common roots.

Personnel/Skill Shortfall

At the national level, there is not an adequate supply of the type of personnel needed for the software life-cycle support task. Evidence of this can be found in the trade journals (e.g., Electronic News), was

[1] Refs. 2-6, 8.

Table 8
MAJOR PROBLEM AREAS FOR EMBEDDED SOFTWARE LIFE-CYCLE SUPPORT

Personnel/Skill Shortfall
Managerial/Organizational Anomalies
Inadequate Documentation
Unavailable Data and Metrics
Rapid Pace of Technological Advance
Failure to Design for Testability

predicted by the COMTEC 2000 study, and is supported by a survey undertaken by the Simulator SPO, which reported major recruiting problems across all ALCs.

Another part of the problem is the increased private sector competition for the specialties needed for software life-cycle support. Moreover, the Simulator SPO noted that, because of the erosion of benefits, experienced ATD personnel are leaving the service to take jobs with private industry--receiving, in some cases, raises in pay and benefits amounting to as much as 50 percent. For example, Ogden ALC's MMEC organization indicated that $1000 to $4000 annual raises are being offered to experienced GS-12 engineers by local computer companies.

Many managers to whom we spoke were alarmed at the rate at which the Air Force has been losing its specialists in computer-related fields. An examination of enlisted and officer continuation rates

confirms this perception.[2] Figure 11 depicts enlisted continuation rates for 1977 and 1979 for the Training Device and Computer Systems career fields, and for all Air Force Specialty Codes (AFSCs) as a whole.[3] These results confirm that the Air Force continuation rate for experienced personnel in the Computer Systems career field is indeed lower than that for all AFSCs as a whole, and there has been some reduction in the continuation rate in this field between 1977 and 1979.

More dramatic changes have occurred in the Training Device career field. In 1977, continuation rates in this field were higher than for all AFSCs as a whole; but by 1979, continuation rates for experienced personnel had slipped to about half the levels of two years earlier. Loss rates in the more skilled enlisted grades for the Training Device and Computer Systems career fields exceeded overall loss rates by 1979 (see Fig. 12).

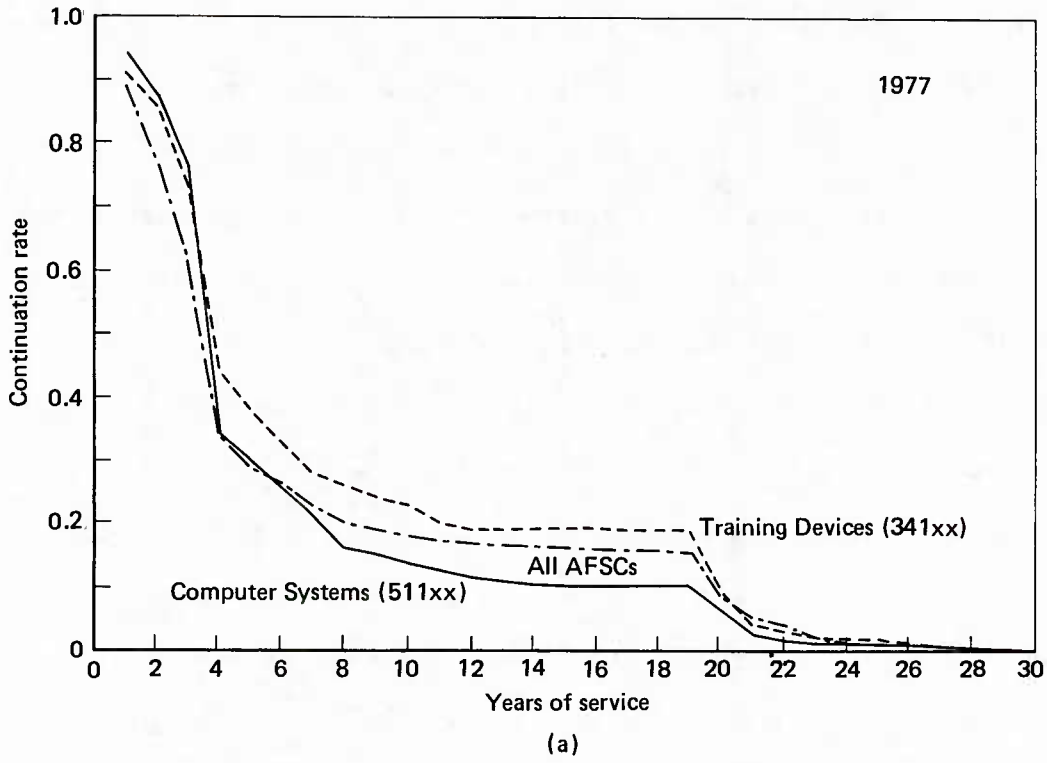
Trends in officer continuation rates are similar to those for enlisted personnel when non-rated line support personnel in the Computer Technology career area (51xx) are compared with total losses of line support personnel.[4] Figure 13 indicates continuation rates are lower for the Computer Technology career area than for all line support personnel and declined between 1978 and 1979.

---

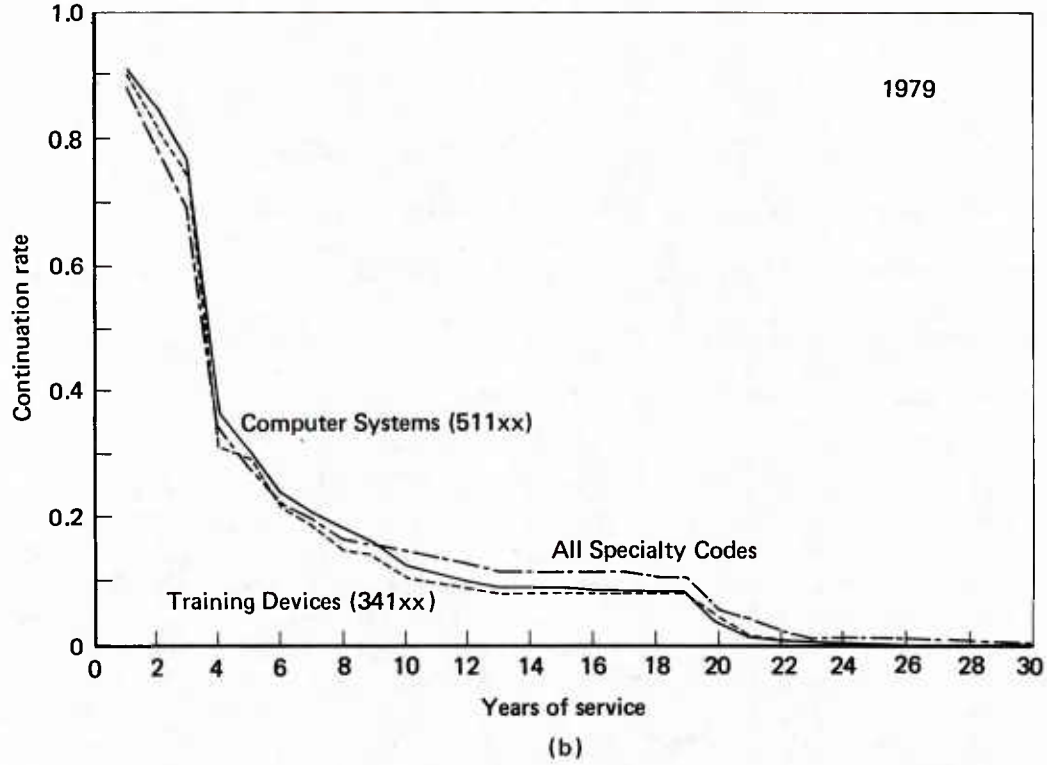
[2] Enlisted data are AFSC Year-Grade Rate History for AFSC 511xx, and Total Enlisted Loss Rates (Historical, by year) prepared by AFMPC/MPCDWQ1. Officer data are Loss Analyses for 51xx non-rated and for total Line Support categories, also prepared by AFMPC/MPCDWQ1.

[3] Note that 511xx fails to cover some computer-relevant AFSCs. Probable effect: the figures shown are conservatively biased.

[4] Non-rated omits pilots, navigators, and certain other individuals from the sample on the assumption that their loss rate stems from a different set of circumstances than for non-rated personnel. The Line Support category omits, for similar reasons, physicians, attorneys, dentists, clergymen and similarly classified individuals.



(a)



(b)

Fig. 11 — Enlisted continuation rates in 1977 and 1979

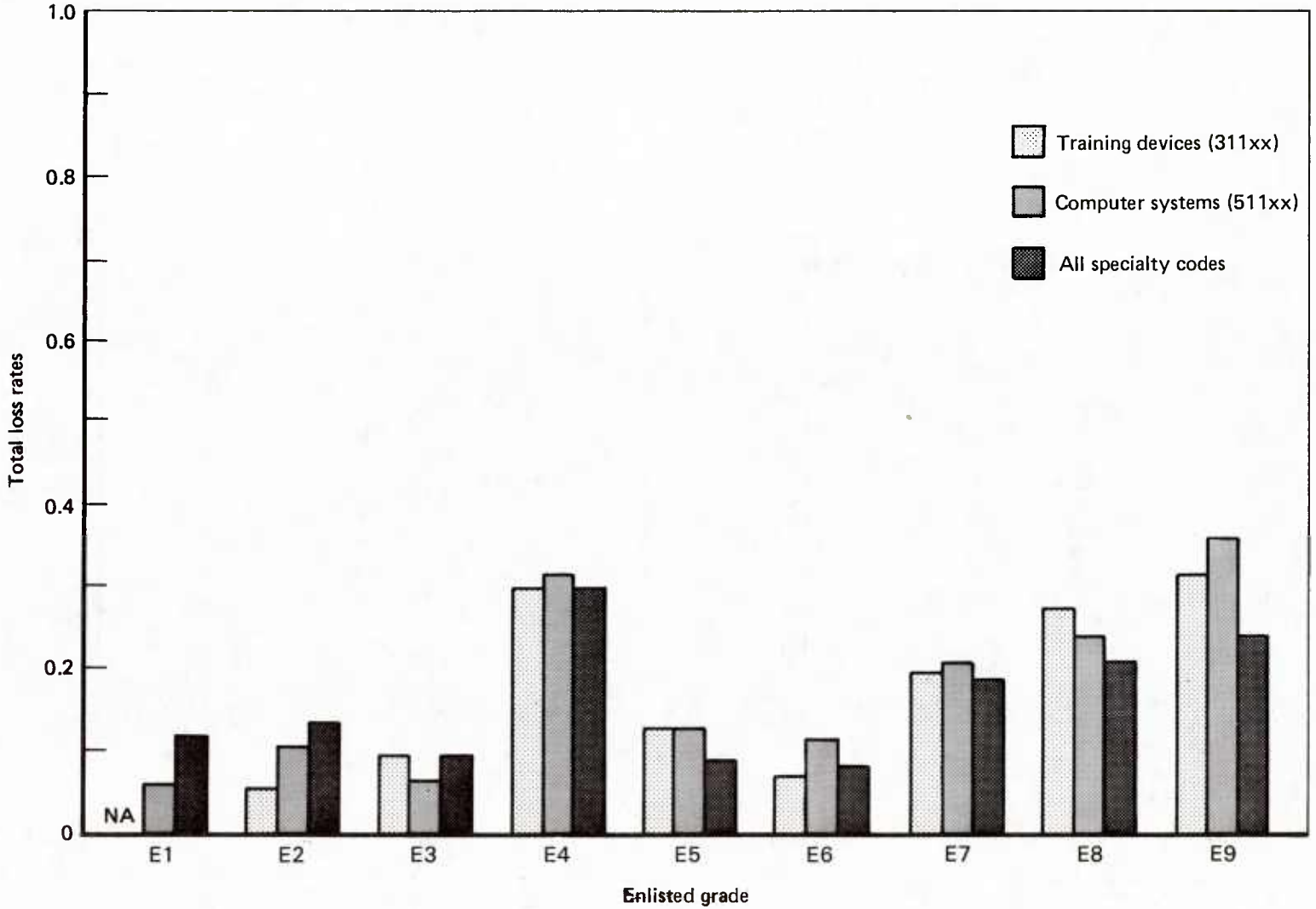
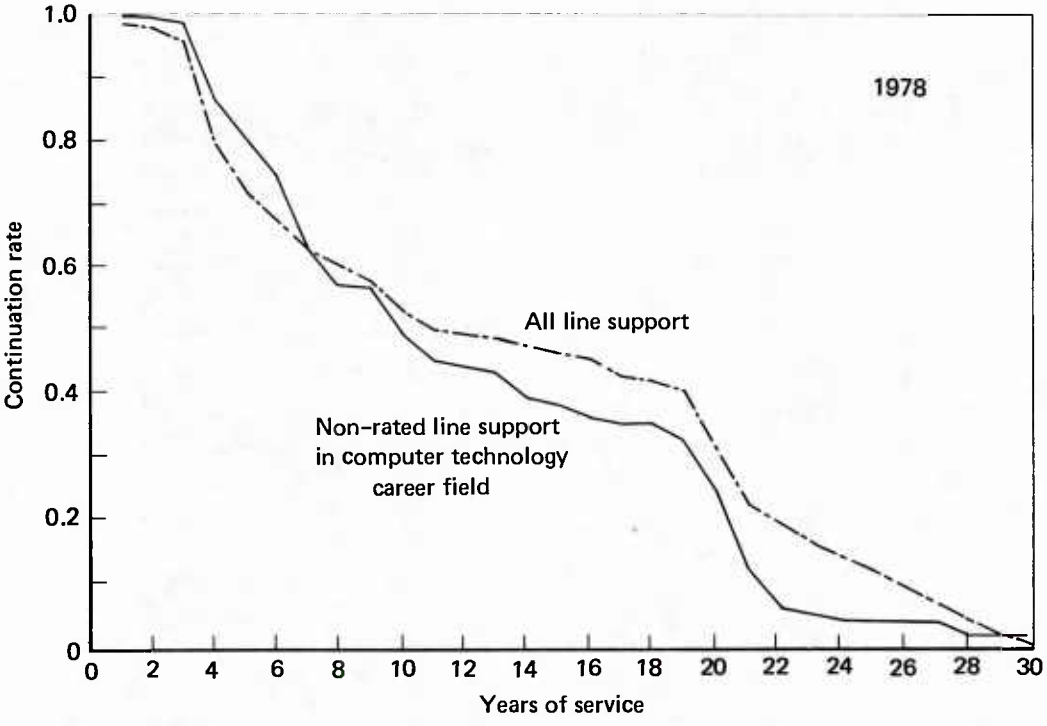
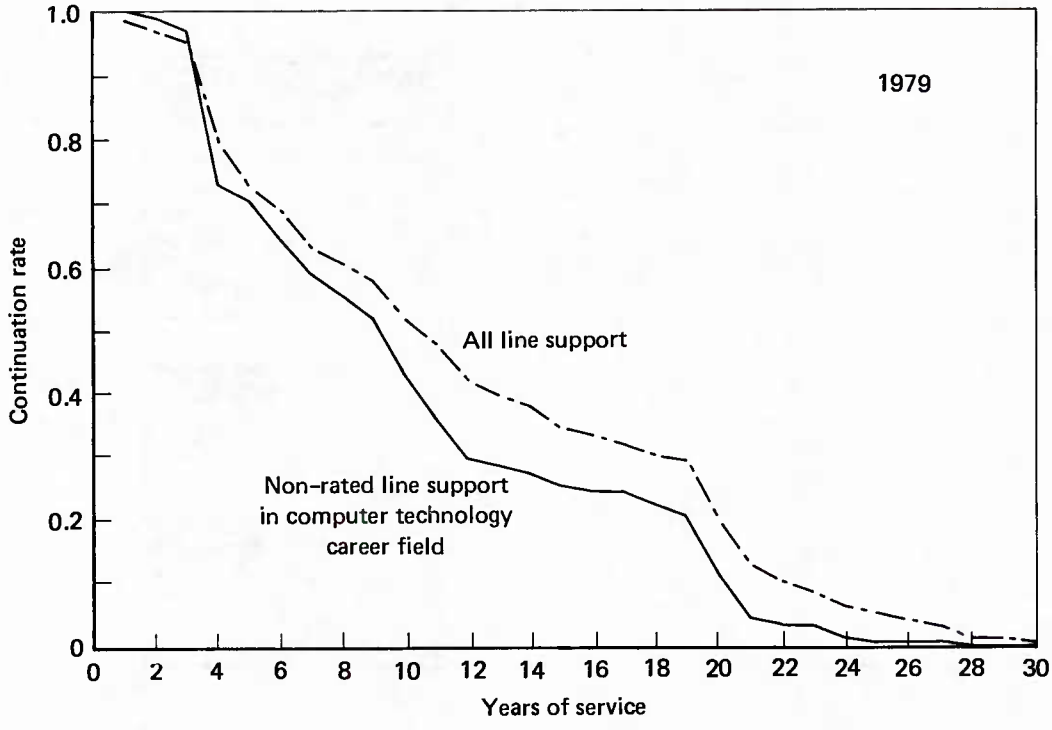


Fig. 12 — Loss rates by enlisted grade in 1979



(a)



(b)

Fig. 13 – Officer continuation rates: 1978 and 1979

Another aspect of the personnel problem has to do with current policies--for example, the standard three year tour of duty for Air Force officers. The B-52 Offensive Avionics System SPO had about 32 man-years of software relevant experience (in six people) when it was reorganized in 1977. By 1979 all of these people (two AF and four civilians) had either rotated out of the SPO or had left for positions elsewhere. Officers who were reassigned as a matter of policy had built considerable experience in the software technical management area, and their loss negatively affected the SPO's operational capability. Fortunately for this particular SPO, a technical assistance contract buffered the blow. Most SPOs, however, are not so well backed up, and the standard rotation policy is starting to receive the questioning attention it deserves in the face of complaints of diminished capacity.

Besides contracting to cover some unfilled manpower slots, the most obvious Air Force response to the personnel/skills issue has been to improve existing educational programs in the computer resources area or implement new ones. Cooperative programs with local colleges, more and more formal on-the-job training and in-house educational programs at the individual ALCs, an expanded Air Force Institute of Technology curriculum, and Participation with Industry programs are some examples we encountered. These initiatives are encouraging, but they are new and it is therefore difficult to evaluate their long-term effects.

ALC personnel, especially in the Materiel Management organizations, are also being influenced by AFLC's recently completed PACER SPAN study, which was intended to suggest means for command-wide improvement in efficiency and economy based on more systematic position management.

This involved developing criteria for determining appropriate average grade levels both within a given organization and across a number of organizations performing similar tasks (e.g., ALCs). PACER SPAN's recommendations are currently being implemented and are expected to be fully carried out in two years. To date, it has resulted in increased average grade levels for two ALCs, allowing them, for example, to provide increased promotion incentives for civilian engineers. Two of the remaining three ALCs, however, have had their average grade levels reduced, and the third stayed about the same.

Managerial/Organizational Anomalies

A number of support organizations indicated that their effectiveness depended heavily on the weapon system SPO's early attention and assignment of resources to support planning. San Antonio ALC, for example, noted that a weapon system SPO is often reluctant to allocate scarce resources early on to permit adequate planning for the production of, for example, Test Requirements Documentation. Warner Robins personnel claim further that the adequacy of all test documentation depends crucially on the weapon system SPO's perception of its priority with regard to other budget items.

In cases where SPOs have invited ALCs to participate in early planning exercises, the ALCs point out that they are poorly staffed for such efforts. Typically, ALC staffing assignments are concurrent with or follow (but do not anticipate) the assignment of system or item support responsibility. In the past, this has frequently occurred toward the end of the development cycle, and it is considered too late to enable adequate planning for software support capabilities.

The Support Equipment SPO noted that the personnel it supplies to weapon system SPOs to aid in support equipment decisionmaking are influential only at the pleasure of that SPO. They feel that good advice is sometimes ignored by program offices that may relegate software support planning to a secondary echelon when they are faced with scarce resources and the resultant need to prioritize.

The Deputy for Avionics Control (ASD/AX) is another organizational entity with influence over software support planning. Although established in 1977, this office until recently had little of its intended influence on weapon system computer resources planning for at least two reasons. First, weapon system SPOs have resisted viewing AX as other than an organizational equal and SPOs could make a case that AX recommendations were non-binding. Second, until November of 1979, AX had only minimal expertise in computer technology. The situation has changed, but the effects of the change have yet to be analyzed.

The Air Force has taken several actions that are expected to ease some of the problems. Workshops, studies, and planning conferences[5] contribute to a better understanding of the problems and to incrementally improved solution alternatives. The advent of such organizations as Acquisition Logistics Division and Deputy for Avionics Control herald a new appreciation of acquisition-support dependencies. As these organizations accumulate experience and expertise, there should be gradually improved communications, coordination, and cooperation between AFSC and AFLC.

---

[5] Refs. 1, 19-21.

Inadequate Documentation

Inadequate test requirements documentation was the most frequent criticism of the ALCs we visited. According to people at San Antonio ALC, typical test requirements documentation for a system might cost from $7 to $25 million.

The F-15 program shows how inadequate documentation can cause difficulties. Because piece-part documentation was not procured, automatic testing of the armament control system must be performed at the subsystem level rather than at the card level. This problem came to light at Bitburg, Germany, when the aircraft's built-in test equipment indicated a fault in the armament control system that could not be replicated by the automatic test equipment. In another instance the F-15 SPO elected not to procure documentation for inertial navigation system depot-level test software, saving at the time approximately $50 million. The decision was made in the face of uncertainty as to whether the system was to be organically maintained. The uncertainty has now been resolved in favor of the organic approach, and circumstances have changed to the extent that it is more cost-effective to reacquire entirely new depot test software and its complete documentation package instead of attempting to document the existing test software.

A survey conducted by Simulator SPO personnel found that documentation quality and quantity headed the list of software support problems reported by ALCs and Development Engineering Prototype sites. They note, for example, that a contractor may deliberately underestimate documentation costs with the intent of distributing them across other elements of the acquisition package. Then, should schedule or budget

pressures make it necessary, the contractor can elect not to produce that documentation and avoid any substantial effect on profit.

Documentation accuracy problems on past programs have moved the manager of the F-16 simulator program to institute formal reviews of contractor documentation planning and production every six weeks throughout the contract performance period. Moreover, to improve simulator software documentation, the Air Force has arranged to sit in on the contractor's internal design reviews in addition to normal participation in the currently mandated formal review structure.

Besides rather thin guidance for software support documentation and discretionary enforcement of existing standards, the broad range of these problems--cost, accuracy, timeliness, completeness, contracting procedures, and Air Force decisionmaking--suggests what may be another of their common sources: an incomplete grasp, by many of those involved, of the kind of information needed by the various groups with software related responsibilities. That is, the development, operational, training, and support communities have overlapping but non identical information requirements, a fact only partly addressed by current documentation guidance and standards. Recent acquisition programs, recognizing potential problems in this area, have tended to experiment with certain aspects of the documentation issue, primarily delivery schedules and adherence to completeness standards. Requirements of information scope, depth, and format as they may vary across different audiences are, with few exceptions, ignored.

The Air Force has been hard at work on this set of problems for some time, and several recent initiatives promise to improve the software documentation picture. The creation of AFSC/AX and its

supporting regulation (AFR 800-28) is one such initiative. Its influence on the development and enforcement of informed standards should guide further improvements. The in-progress update to MIL-STD 881A[6] is another positive step.

Draft AFLC Regulation 800-21 goes into considerable detail on AFLC's role with respect to supporting embedded computer resources. It addresses all categories of defense system software describing policies, concepts of operation, and responsibilities in each area, and providing suggested documentation formats for more effective communication among participants.

The Computer Program Identification Numbering (CPIN) System is a somewhat controversial initiative that is intended to standardize the means of identifying computer programs and related documentation media. This seems like a worthwhile idea, but several potential pitfalls require attention, the most serious of which is the apparent softness in the definition of "computer program." In its Terminology section, Draft AFLC Regulation 800-21 defers to the definition contained in AFR 800-14 (see Vol. 1, Attachment 1). In later references to CPIN, however, 800-21 implies an equivalence between computer program and Computer Program Configuration Item (CPCI). Apparently the CPIN system is addressed to the CPCI level of software design, and this could lead to problems with those who treat CPCIs as made up of smaller, logically and functionally consistent entities known as Computer Program Components. Other objections are that CPIN is a complicating addition to an already overly complex documentation tree and that it is unnecessary because

---

[6] MIL-STD 881A establishes criteria governing the preparation and employment of Work Breakdown Structures for use during the acquisition of designated defense material items.

there are already adequate means for identifying specific program modules.

One of the major objectives of the modular automatic test equipment (MATE) program is to develop a general framework for acquiring and supporting ATE. We would expect one of the primary elements of any such framework to deal explicitly with ATE support documentation requirements. To the extent that this is true, and that the MATE program reaches fruition as planned sometime during 1984, it may provide a model for improved documentation practices for other software communities to follow.

A large problem in contracting for software documentation is knowing what to ask for. There are no standard Data Item Descriptors for software. Each contract tends to invent its own descriptors, or the software portion of the Contract Data Requirements List is developed on an ad hoc basis. Neither approach is inherently bad, but the resulting proliferation of techniques and differing requirements works against the development of a more uniform and manageable grasp of the problem. MITRE Corporation, under an ESD contract, has recently produced a draft software Data Item Descriptor that is currently under test as an Air Force standard.[7]

Finally, the AFTEC-sponsored IOT&E software maintainability evaluation effort is an in-house study of the adequacy of documentation for software support. Oklahoma City ALC has responded to questionnaires dealing with several aspects of support documentation: descriptiveness, modularity, simplicity, and instrumentality (intended as an indicator of

---

[7] Ref. 22.

the degree to which documents facilitate actual support tasks). All these characteristics are noted in MIL-STD 483.[8]

Unavailable Data and Metrics

Few SPOs or ALCs seem to collect or to request contractors to report detailed software data on any aspect of software acquisition or support. The last Joint Logistics Commanders Software Workshop[9] and the recent Industry/Joint Services study of software testing[10] are the latest in a long line of studies calling attention to this problem. A few of the organizations visited did attempt to collect data on their particular software operations; but the methodologies used, the kinds of data collected, and the meanings assigned to them vary so widely and are so controversial that they hinder the emergence of generally relevant principles. The data that are collected are idiosyncratic and are therefore vulnerable to a claim of irrelevancy for other programs.

The Industry/Joint Services study mentioned above is a composite assessment of the current state of automatic testing capabilities across the services as they pertain to the affordability of new weapon systems. The final report noted that a lack of suitable data precluded any quantitative definition of benefits to be expected from any of its recommendations, including those related to ATE system software.

---

[8] MIL-STD 483 establishes requirements for configuration management. It also establishes supplementary requirements for identification, interface, control, audits, control of engineering changes, and management reports, records, and plans.

[9] Ref. 7.

[10] Ref. 23.

The Simulator, F-16, and B-52 SPOs made similar observations. Program offices infrequently account in any detail for their software-specific activities because, given complex and multifaceted personnel roles, it is difficult to know how to assign particular segments of work to predominantly software issues. Similarly, contractors are usually not required to report detailed software development statistics because their diverse internal estimation and accounting schemes render the data useless for the comprehensive planning needs of the SPO. Contractors claim further that collecting data of the kinds desired would be prohibitively expensive, an argument not lost on resource strained program offices.

More to the point, there is not yet a well-defined and generally accepted set of data elements that, if collected, would provide an adequate basis for software decisionmaking. Even more basic questions related to the feasibility and cost-effectiveness of collecting such data remain unanswered. In this regard, the software engineering community has, despite good progress over the last decade,[11] so far failed to provide an acceptable set of metrics upon which to base data collection.

The result has been a lack of rigorous analytical bases for many of the multi-million dollar software decisions made by the Air Force and others. There are correspondingly strong suspicions that things would be better if such decisions were based on the results of analyses supported by regularly scheduled data collection efforts, themselves based on a generally accepted set of software metrics, standard collection instruments, validity checks, and analytic packages.

---

[11] Ref. 24.

The Air Force has approached this set of problems from several directions. The most apparently direct approach is the Data and Analysis Center for Software, established in 1978 at Rome Air Development Center.

As far as we could tell, this facility has yet to affect the software support community, mainly because benefits must follow the development and analysis of a large amount of data, which takes considerable time even in environments where "what to collect" is well understood.

The road maps, study recommendations, and data collection issues raised at the last two Avionics and Armament Planning Conferences have been of some indirect help, by calling high level attention to the problem. It remains to be seen what direct improvements will emerge from the initiatives fostered by these sources.

Relevant here as well as to the documentation issues raised earlier is the ESD-sponsored effort to develop and test a standard Data Item Description (DID) list for software. To the extent such standards achieve broad acceptance they should have far-reaching effects on software contracting, on in-progress review of software development, on testing procedures, and on support capability. However, developing standard software DIDs without concurrent attention to their logical relationship with metrics would unduly limit the benefits available from the DID and could prolong the development and complicate implementation of software metrics later considered.

Rapid Pace of Technological Advance

Technology advance is usually associated with new and improved capabilities and performance at generally lower costs. There is understandable pressure to move as rapidly as possible to the latest technology from both the Air Force and the contractor communities--the former from the view of improved operational capability, the latter from the view of increased economies of production. However, the pace at which advances are made make it quite difficult to keep organizations, policies, and skills up to date. In many programs the acquisition cycle is so long that hardware and software are obsolete before the system is placed into operation.

The primary attack on this problem seems to involve greater use of commercial equipment. It is hoped, by the Simulator SPO, for example, that that approach will shorten the line (dominated by the expensive and lengthy "militarization" process) between vendors and users, thus easing some of the problems mentioned above. This idea has its counterpart in software. Vendors continually update their software to match new or updated hardware. It may be more cost-effective and timely to acquire vendor developed software and commercial hardware than to contract separately for it. Paradoxically, this can lead both to increased dependency on contractor support in cases where Air Force skills lag the technologies requiring support, and to heavier dependence on organic support in cases where the contractor no longer supports what are considered old technologies. The Simulator SPO is looking into these ideas. Greater utilization of commercial equipment opens the door to data rights issues--already a problem with software contracting--and it

could well raise serious questions about the adequacy of organic support for non-militarized hardware.

Ogden ALC reports that the growing complexity of line replaceable units in some modern systems almost precludes the use of Automatic Test Program Generators (ATPG).[12] For example, some newer technologies tend to blur the distinction between hardware and software. This complicates the testing environment, making it necessary, in some cases, to rely on manual procedures. Obviously, this diminishes the utility of ATPG technology. It also increases costs, because manual generation is both more expensive and less efficient. The same problem complicates the job of Development Engineering Prototype sites, for example, when they must modify object-level software that is increasingly hard to distinguish from its highly integrated hardware counterpart. At another level, AFLC/ALD notes that current procedures for assigning ALC system or item support responsibility by device nomenclature or category run into trouble with increased software-hardware integration.

One of the threads connecting many aspects of the problem of rapid technological advance seems to be the conflict between gaining the advantages of advancing technologies and minimizing problems associated with the proliferation of new and different implementations (hardware and software) of similar functions. Part of the approach to resolving this conflict is the application of appropriate standards, a complex issue in itself. Standards are needed that permit efficiencies in training and procurement without locking the Air Force into long-term dependence on transient technological characteristics.

---

[12] Automatic Test Program Generators are computerized tools to assist the test engineer in generating test software for digital components. These tools primarily generate programs for testing printed circuit boards.

Perhaps the most broadly relevant Air Force initiative in this area is the creation of the Deputy for Avionics Control within Logistics Command (AFLC/AX). Under the guidance set forth in AFR 800-28, this office is chartered to oversee and carry out Air Force avionics policy. Among other things, this office is concerned directly with the identification, development, promulgation, and enforcement of Force-wide standards; for example, in the areas of instruction set architectures (MIL-STD-1750), interfaces (MIL-STD-1553B), and support equipment (AFLC plans for standardizing Avionics Integration Support Facility design).

Failure to Design for Testability

Failing to provide adequately for testing requirements in the design of new or updated equipment can add up to 100 percent to testing costs.[13] From our interviews, the greatest concern seems to be at the printed circuit board level. In a few instances some of the newest printed circuit boards already exceed the testing capabilities of existing automatic test equipment, because of the failure to consider testing during the conceptual phase of system design. Ogden personnel claim that if testability had been designed into printed circuit boards used on the F-15 and F-16 aircraft, the depot repair costs associated with those systems would be 30-35 percent lower than current experience.[14] For the F-16 program, for example, that works out to about $11.7 million per year considering initial development and annual repair costs.[15]

---

[13] Ref. 25.

[14] Ref. 26.

[15] ASD/ENA is working to identify appropriate levels of testability for airborne systems by comparing the extra costs of design in testability against cost savings in maintenance.

Aside from the direct effects on equipment testing and on operational readiness, test program development itself becomes more vulnerable to problems because ATE software must be designed to compensate for inadequate test design. The resultant increase in software complexity contributes to error-prone operational characteristics.

Increasing circuit densities and more complex interfaces are less tolerant of inadequate test planning. If testability does not soon become a major factor in design, some devices will be untestable as produced. To the best of our knowledge, an Air Force response to this issue has been slow in coming. Indeed, before a direct attack can be mustered on this problem, some more fundamental problems must be resolved. These have to do with our currently incomplete grasp of test criteria and metrics, of testing information/documentation requirements, and of testing technology in general.

Other Problem Areas

When spare storage and/or processing capacity is insufficient, additions or modifications to system capabilities--a large part of the software support task--must be made at the expense of lower priority capabilities. The problem itself has several characteristics. First, where changes require more capacity than is available, prioritization of existing system capabilities is a prerequisite to software cannibalization. Some low priority capability is lost--in effect it represents a hidden additional cost for the desired change. One is tempted to suggest that such prioritization and deletion of, say, the lowest two items on the list are the last step before system delivery in

anticipation of future needs. That, however, begs the question and misrepresents what is a useful solution in some cases.

Often, however, the cannibalization approach is not feasible. Earlier actions might have exhausted the "low priority" capabilities, or the contemplated change requires capabilities that cannot be completely carried out on the existing hardware-software suite. Substantial additional costs may then be incurred for new hardware capable of supporting existing and new capabilities--including some provision for spare capacity for future modifications.

One reason for not rigidly enforcing the requirement of a specific amount of space capacity upon system delivery is that system requirements undergo frequent changes during the development cycle, necessitating corresponding "on the fly" changes to partially designed and implemented software. When a single change uses, for example, only 5 or 10 additional storage units, it is easy to assume away any effect on what appears to be ample spare capacity. In this way successive changes, some smaller, some larger, eat away at a smaller and smaller reserve until almost none remains, thus furnishing the support community with a built-in problem at Program Management Responsibility Transfer. This aspect of the problem has been discussed for years, with little effect beyond frequent calls for rigidly enforced design baselines and tighter configuration control techniques. In fact, part of the problem seems to turn on (1) the general lack of understanding of software and the levers that influence its development and support, and (2) an acquisition philosophy that emphasizes technical sophistication as a hedge against decreased numbers of systems procured.

Another driver involves the degree to which future changes are anticipated during software development. Given that spare capacity represents a cost, there is a tendency toward conservatism. Looking beyond well known and certain short-term changes is infrequent under these conditions. The fact that other potential changes might be visible enough to warrant some discussion, and perhaps some accommodation, should be weighed against the certain short-term acquisition effect of that outcome.

As weapon system software increases in functional and interface complexity, higher levels of automation will be required for its development, testing, and support. The Air Force, however, seems not to be taking full advantage of the progress in software tool development that is taking place in the commercial sector. Apparently no one is actively tracking this area of software technology for potential early benefits to weapon system software development and support. At the least our interviews suggest that, even if some organization is tracking the area (e.g., the Data and Analysis Center for Software mentioned earlier) benefits have yet to begin accruing to the software acquisition and support communities. This leaves the job to the individual SPOs and ALCs and, although that is not ineffective, it contributes to a tendency to approach force-wide issues in a piecemeal fashion. The result may be missed opportunities for cost avoidance, and the possibility argues strongly for a study of at least that aspect of the software tools issue.

EMERGING ISSUES

Microprocessors/Firmware

As we have seen it used in the Air Force, the term "firmware" tends to refer primarily to microprocessor devices. Generally the term refers to software that is put into some form of "Read Only Memory" and therefore not subject to change by the computer itself. The danger we see is that as these devices find their way into more and more defense systems we stand to repeat many of the same errors that still plague the acquisition and support of larger scale processors. The Air Force already has a problem responding to the pace of technological advance for equipment that is not based on firmware, and unless means are found to better cope with such moving targets, any major future commitment to firmware could find the Air Force unprepared to support these systems.

There is some appreciation of the data-rights issue as it applies to firmware. Originally a problem with software, the need for enlightened contracting will be more crucial for firmware-based systems. For example, unless explicit attention is given to the need for data that will permit organic support for these devices, that option could be rendered unavailable without increased expenditures for additional documentation. Moreover, the nature of the required information is largely unknown at present, and the standards questions are only now being identified.

A start in the right direction is AFSC white paper[16] that proposes a number of definitions and calls for some changes to the 1977 supplement to AFR 800-14 aimed at improving consideration for

---

[16] Ref. 27.

microprocessor acquisition and support in current policy. The recommendations reflect a direction of thought that should be encouraged and taken up by a larger number of individuals in the planning community.

Very High Speed Integrated Circuits (VHSIC)

This issue encompasses the technology addressed in the issues discussed above and presents some of the same problems--e.g., data rights. It differs in scope, however; here the topic is the whole of integrated circuit technology, not just the microprocessor/firmware relevant applications. Also, just as firmware is considered as having a data processing emphasis, so the VHSIC program seems to emphasize signal processing applications.

Differences in embedded software implementations (e.g., the use of higher order rather than assembly language) influence the type of VHSIC device that is appropriate to a given application, and the nature (speed, waveform, etc.) of the interfaces required. Different users have different preferences for language standards. The choice of language for software-aided chip design tools will strongly influence not only the design efficiency of the resultant integrated circuits, but also the management of the program itself.

Both of the issues just discussed promise changes to technology and subsequent changes in system and component architectures that will impose new requirements in the areas of software acquisition guidelines, testing concepts, and support techniques. The sooner the problems discussed earlier in this section are remedied and the sooner they include attention to these emerging issues, the less likely it is that they will be added to the list of in-place problems.

BIBLIOGRAPHY

1. United States Air Force Avionics Master Plan, The Deputy for Avionics Control (ASD/AX), July 1979.

2. DoD Weapons System Software Management Study, The Johns Hopkins University Applied Physics Laboratory, APL/JHU SR 75-3, Maryland, June 1975.

3. Computer Technology Forecast and Weapon System Impact Study (COMTEC-2000), Hq AFSC, TR-78-03, July 1979.

4. S. M. Drezner et al., The Computer Resources Management Study. The Rand Corporation, R-1855/1-PR, April 1976.

5. Embedded Computer Resources Support Planning for the 1980s (ESP-80), for AF/ALD Air Force Logistics Command, April 1980.

6. S. Glaseman and M. R. Davis, Software Requirements for Embedded Computers: A Preliminary Report, The Rand Corporation, R-2567-AF, March 1980.

7. Report of the Panel on Software Acquisition/Development Standards, Joint Logistics Commanders Software Workshop, Monterey, California, April 3-5, 1979.

8. DoD Weapons System Software Acquisition and Management Study, MITRE Corporation, May 1975.

9. Project Pacer Flash - Executive Summary and Final Report, Air Force Logistics Command, Wright-Patterson Air Force Base, Ohio, September 1973.

10. Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980s (CCIP-85), Vol. I, SAMSO/XRS-71-1, February 1972.

11. Minutes of the Modeling Subpanel, Second Annual Avionics Planning Conference, Colorado Springs, Colorado, October 31-November 7, 1978.

12. Air Force Avionics Laboratory Systems Avionics Division Advanced Systems Avionics (ASA) Workshop with Industry, Wright-Patterson Air Force Base, Ohio, June 1979.

13. Proceedings of the Aeronautical Systems Software Workshop, Dayton, Ohio, April 2-4, 1974.

14. F-111 Software Maintenance Requirements and Costs, Dynamics Research Corporation, E-5224U, October 1979.

15. R. K. E. Black et al., "ECS Software Production Data," Boeing Computer Services, Inc., RADC-TR-77-116, March 1977.

16. F-111 A/E Digital Bomb-Nav System Software Analysis, Battelle-Columbus Laboratories, Technical Report AFAL-TR-79-1043, November 1978.

17. "Average Cost of Military and Civilian Manpower in the Department of Defense," Office of the Assistant Secretary of Defense (Comptroller), December 1977.

18. J. Ferrell, "Logical Planning for Automatic Test Equipment Software Requirements," Computers in Aerospace Conference, Los Angeles CA, October 31 - November 2, 1977.

19. "National Software Works Technology Demonstration Plan for Air Force Logistics Command Applications" (Draft), AFLC-RADC, October 1979.

20. E. F. Reed, Avionics Software Development Experiences from the B-1 Program, The Boeing Aerospace Company, November 1977.

21. J. L. Wilson and R. W. Morton, "A Methodology for Analysis of Alternatives in the Acquisition and Support of Automatic Test Equipment Software," Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, Report No. GSM/SM/74D-8, December 1974.

22. W. E. Byrne et al., "Draft Data Item Description (DID) for Software Acquisition Resource Expenditure (SARE) Reporting," MITRE Corporation, Working Paper WP-21975, Vol. 1, December 1978.

23. Industry/Joint Services Automatic Test Project, Final Report, April 1980.

24. B. W. Boehm, "Software Engineering," IEEE Trans. on Computers, Vol. C-25, No. 12, December 1976.

25. G. Pedersen, Reducing ATE Test Software Development Costs, Hill AFB, Utah (unknown date).

26. Design for Testability, Briefing prepared by Ogden Air Logistics Center, Hill AFB, Utah, December 1979.

27. R. J. Sylvester, "Revised White Paper on AFSC Microprocessor Policy," ASD/EN, Wright-Patterson AFB, Ohio, January 1980.

28. Minutes of the Acquisition Management Policy Subpanel, Armament and Avionics Planning Conference, Nellis AFB, Nevada, October 15-19, 1979.

29. Air Force Policy on Avionics Acquisition and Support, Air Force Regulation 800-28, Department of the Air Force, September 1978.

30. Policies, Responsibilities, and Procedures for Obtaining New and Improved Operational Capabilities, Air Force Regulation 57-1, August 1971.

31. OC-ALC Embedded Computer Systems (ECS) Integrated Software Support Plan, Vols. I through IV, Oklahoma City Air Logistics Center, January 1979.

32. R. J. Arceneaux and G. E. Farschman, "An Assessment of Relevant Decision Making Factors for Organic versus Contract Maintenance Options on USAF Flight Simulators," Thesis, DCC Report No. LSSR 7-77B, September 1977.

33. USAF Aeronautical Systems Operational Software Maintenance Study, AFSC/ASD, Report No. ASD-TR-73-46, October 1973.

34. "Definition of an Approach to Establishment of an F-15 Avionics Software Support Capability," AFSC/ASD, Technical Report ASD-TR-74-30, July 1974.

35. Computer Program Maintenance, The Boeing Company, D180-22812-1, December 1977.

36. Contracting for Software Acquisition, The Boeing Company, D180-24701-1, January 1979.

37. R. House, "Comments on Program Specification and Testing," Communications of the ACM, Vol. 23, No. 6, June 1980.

38. F-16 Multinational Computer Resources Integrated Support Plan (CRISP) - F-16 Operational Flight Program, F-16-1001, February 1979.

39. F. T. Kimball, "The Transition of F-111 Intermediate Level Automatic Test Equipment Software Responsibility from AFSC to AFLC: A Case Description and Analysis," Thesis, Air Force Institute of Technology, Report No. GSM/SM/73-15, September 1973.

40. Management Guide to Avionics Software Acquisition, Vol. 11: Software Acquisition Process, Logicon Corporation, ASD-TR-76-11, June 1976.

41. Management of Automatic Data Processing Systems, Air Force Regulation 300-2, February 1975.

42. Management of Computer Resources in Systems, Air Force Regulation 800-14, Vols. I and II, Department of the Air Force, September 1975.

43. Specification Practices, Military Standard (MIL-STD) 490, October 1968.

44. Military Specification - Digital Computational System for Real-Time Training Simulators, USAF, MIL-D-83468, December 1975.

45. Department of Defense Directive 5000.1, Major System Acquisitions, January 18, 1977.

46. Measuring and Reporting Software Status, The Boeing Company, D180-24728-1, August 1978.

47. Proceedings of the IEEE 1980 National Aerospace and Electronic Conference, Dayton, Ohio, May 20-22, 1980.

48. OMB Circular No. A-109, Major Systems Acquisitions, Office of Management and Budget, April 5, 1976.

49. Computer Program Documentation Requirements, The Boeing Company, D180-20675-1, July 1977.

50. Project Test Report for FB-111A Operational Flight Program FB-16, Sacramento Air Logistics Center, Report No. 79-04, March 1979.

51. R. L. Glass, "Real-Time: the 'Lost World' of Software Debugging and Testing," Communications of the ACM, Vol. 23, No. 5, May 1980.

52. Department of Defense Directive 5000.2, Major System Acquisition Process, January 18, 1977.

53. Software Engineering and Management Plan, Vol. II, Air Force Systems Command, AFSC/XRF, March 1976.

54. Operational Requirements: Statement of Operational Need (SON), Air Force Regulation 57-1, June 1979.

55. Software Quality Assurance, The Boeing Company, D180-24700-1, January 1979.

56. Requirements Specifications, The Boeing Company, D180-20676-1, January 1979.

57. Avionics Operational Flight Program Integrated Support Plan, prepared for Warner Robbins ALC by TRW Defense and Space Systems Group, Report No. 31073-6041-TU-00, January 1978.

58. Bruce A. Vendt, "Software Support for the F-16 Avionics Computers," Thesis, GSM/SM/75-D-23, December 1975.

59. R. S. Ziernicki, "Avionics: The Road Ahead," Air Force Magazine, July 1979, pp. 67-75.